The Journal of Machine Learning Research

Volume 8
Print-Archive Edition

Pages 1–1392

Microtome Publishing
Brookline, Massachusetts
www.mtome.com

The Journal of Machine Learning Research
Volume 8
Print-Archive Edition

The Journal of Machine Learning Research (JMLR) is an open access journal. All articles published in JMLR are freely available via electronic distribution. This Print-Archive Edition is published annually as a means of archiving the contents of the journal in perpetuity. The contents of this volume are articles published electronically in JMLR in 2007.

JMLR is abstracted in ACM Computing Reviews, INSPEC, and Psychological Abstracts/PsycINFO.

JMLR is a publication of Journal of Machine Learning Research, Inc. For further information regarding JMLR, including open access to articles, visit http://www.jmlr.org/.

JMLR Print-Archive Edition is a publication of Microtome Publishing under agreement with Journal of Machine Learning Research, Inc. For further information regarding the Print-Archive Edition, including subscription and distribution information and background on open-access print archiving, visit Microtome Publishing at http://www.mtome.com/.

ISSN 1532-4435 (print)
ISSN 1533-7928 (online)

JMLR
Journal of Machine Learning Research Vol. 8 2007

Journal of Machine Learning Research 8 (2007) 1-33 Submitted 8/05; Revised 7/06; Published 1/07

Nonlinear Boosting Projections for Ensemble Construction

Nicolás García-Pedrajas NPEDRAJAS@UCO.ES
Department of Computing and Numerical Analysis
University of Córdoba
Córdoba, Spain

César García-Osorio CGOSORIO@UBU.ES
Department of Civil Engineering
University of Burgos
Burgos, Spain

Colin Fyfe COLIN.FYFE@PAISLEY.AC.UK
School of Computing
University of Paisley
Paisley, United Kingdom

Editor: Dale Schuurmans

Abstract

In this paper we propose a novel approach for ensemble construction based on the use of nonlinear projections to achieve both accuracy and diversity of individual classifiers. The proposed approach combines the philosophy of boosting, putting more effort on difficult instances, with the basis of the random subspace method. Our main contribution is that instead of using a random subspace, we construct a projection taking into account the instances which have posed most difficulties to previous classifiers. In this way, consecutive nonlinear projections are created by a neural network trained using only incorrectly classified instances. The feature subspace induced by the hidden layer of this network is used as the input space to a new classifier. The method is compared with bagging and boosting techniques, showing an improved performance on a large set of 44 problems from the UCI Machine Learning Repository. An additional study showed that the proposed approach is less sensitive to noise in the data than boosting methods.

Keywords: classifier ensembles, boosting, neural networks, nonlinear projections

1. Introduction

An ensemble of classifiers consists of a combination of different classifiers, homogeneous or heterogeneous, to jointly perform a classification task. Ensemble construction is one of the fields of Artificial Intelligence that is receiving most research attention, mainly due to the significant performance improvements over single classifiers that have been reported with ensemble methods (Breiman, 1996a; Kohavi and Kunz, 1997; Bauer and Kohavi, 1999; Webb, 2000; García-Pedrajas et al., 2005).

A classification problem of K classes and n training observations consists of a set of instances whose class membership is known. Let $S = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots (\mathbf{x}_n, y_n)\}$ be a set of n training samples where each instance $\mathbf{x}_i$ belongs to a domain X. Each label is an integer from the set $Y = \{1, \ldots, K\}$. A multiclass classifier is a function $f : X \to Y$ that maps an instance $\mathbf{x} \in X \subset \mathbb{R}^D$ onto an element of Y.

The task is to find a definition for the unknown function, $f(\mathbf{x})$, given the set of training instances. In a classifier ensemble framework we have a set of classifiers $\mathbb{C} = \{C_1, C_2, \ldots, C_m\}$, each classifier performing a mapping of an instance vector $\mathbf{x} \in \mathbb{R}^D$ onto the set of labels $Y = \{1, \ldots, K\}$. The design of classifier ensembles must face two main tasks: constructing the individuals classifiers, C_i, and developing a combination rule that finds a class label for $\mathbf{x}$ based on the outputs of the classifiers $\{C_1(\mathbf{x}), C_2(\mathbf{x}), \ldots, C_m(\mathbf{x})\}$. This paper is devoted to the first problem, the combination of the classifiers is being done with a simple majority voting scheme.

For more detailed descriptions of ensembles the reader is referred to other reviews: Dietterich (2000b), Webb (2000), Dzeroski and Zenko (2004), Merz (1999), or Fern and Givan (2003).

Techniques using multiple models usually consist of two independent phases: model generation and model combination (Merz, 1999). Most techniques are focused on obtaining a group of classifiers which are as accurate as possible but which disagree as much as possible. These two objectives are somewhat conflicting, since if the classifiers are more accurate, it is obvious that they must agree more frequently. Many methods have been developed to enforce diversity on the classifiers that form the ensemble (Dietterich, 2000c). Kuncheva (2001) identifies four fundamental approaches: (i) using different combination schemes, (ii) using different classifier models, (iii) using different feature subsets, and (iv) using different training sets. Perhaps the last one is the most commonly used. The algorithms in this last approach can be divided into two groups: algorithms that adaptively change the distribution of the training set based on the performance of the previous classifiers, and algorithms that do not adapt the distribution. Boosting methods are the most representative methods of the first group. The most widely used boosting methods are ADABOOST (Freund and Schapire, 1996) and its numerous variants, and Arc-x4 (Breiman, 1998). They are based on adaptively increasing the probability of sampling the instances that are not classified correctly by the previous classifiers.

Bagging (Breiman, 1996b) is the most representative algorithm of the second group. Bagging (after *B*ootstrap *agg*regat*ing*) just generates different bootstrap samples from the training set. Several empirical studies have shown that ADABOOST is able to reduce both bias and variance components of the error (Breiman, 1996c; Schapire et al., 1998; Bauer and Kohavi, 1999). On the other hand, bagging seems to be more efficient in reducing bias than ADABOOST (Bauer and Kohavi, 1999).

Although these techniques are focused on obtaining as diverse classifiers as possible, without deteriorating the accuracy of each classifier, Kuncheva and Whitaker (2003) failed to establish a clear relationship between diversity and ensemble performance.

Boosting methods are the most popular techniques for constructing ensembles of classifiers. Its popularity is mainly due to the success of ADABOOST. However, ADABOOST tends to perform very well for some problems but can also perform very poorly on other problems. One of the sources of the bad behaviour of ADABOOST is that although it is always able to construct diverse ensembles, in some problems the individual classifiers tend to have large training errors (Dietterich, 2000a). Moreover, ADABOOST usually performs poorly on noisy problems (Bauer and Kohavi, 1999; Dietterich, 2000a).

Schapire and Singer (1999) identified two scenarios where ADABOOST is likely to fail:

1. When there is insufficient training data relative to the "complexity" of the base classifiers.

2. When the training errors of the base classifiers become too large too quickly.

Sebban et al. (2002) proposed a stopping criterion for boosting algorithms, and there are other variants (Schapire and Singer, 1999; Webb, 2000; Bshouty and Gavinsky, 2002; Eibl and Pfeiffer, 2005) that try to overcome the drawbacks of the method. Nevertheless, ADABOOST is still likely to fail on noisy problems or when there is not much available data. Unfortunately, these two scenarios are very common in real-world problems.

The margin (Mason et al., 2000) of a real-valued function $f : X \to \mathbb{R}$ on a training instance $(\mathbf{x}, y) \in X \times \{-1, 1\}$ is defined as $yf(\mathbf{x})$, so that if the sign is correct the margin is positive. The size of the margin can be interpreted as an indication of the confidence of the classification. The success of ADABOOST has been partially explained in terms of the "boosting" of the margins that usually is achieved, however this is not the only important factor in its success. Several experiments have shown that ADABOOST is able to perform better than algorithms that are more efficient than ADABOOST in optimising the margins (Breiman, 1999; Grove and Schuurmans, 1998). There are other attempts to explain the good performance of boosting (Rosset et al., 2004).

Alternatively, some works have focused on using different subsets of the inputs, that is, different subspaces, to train the classifiers. Cherkauer (1996) trained an ensemble of classifiers which consisted of 32 neural networks trained using 8 different subsets of input features together with 4 different network sizes. Chen et al. (1997) studied the use of different features for training multiple classifiers in a framework of text-independent speaker recognition. Tumer and Ghosh (1996) used a similar technique to classify sonar signals, but they found that removing just a few of the input features hurts the performance of the classifiers so much that the resulting ensemble had a poor performance.

As a matter of fact, it has been shown that this technique is capable of obtaining a good performance only when the input features are highly redundant (Dietterich, 2000b). Ho (1998) proposed a method for constructing a decision forest by randomly selecting subspaces from the original data set, and reported very good results.

As an useful alternative, evolutionary computation has also been successfully applied to ensemble construction. Zhou et al. (2002) used a genetic algorithm to obtain a subset of an ensemble of classifiers that was able to outperform the ensemble using all the classifiers. Liu et al. (2000) used the concept of *negative correlation* to improve the diversity of a population of classifiers. García-Pedrajas et al. (2005) developed a cooperative coevolutionary method for ensemble construction with excellent results on classification problems. Ortiz-Boyer et al. (2005) used a real-coded genetic algorithm to optimise the weights of each classifier within an ensemble.

In summary, the construction of ensembles aims to fulfil two different objectives: accuracy of classifiers and diversity among them. These two objectives are partially in conflict, as the more accurate the classifiers are the less diverse they must be. Among the different approaches presented above, we can highlight the following techniques to enforce accuracy and diversity:

- Bagging methods sample the training data with replacement to obtain different sets to train the classifiers.

- Boosting methods enforce accuracy and diversity by putting more emphasis on instances that are misclassified by previous classifiers.

- Subspace methods encourage diversity by training the classifiers using different subsets of input features or different projections of the original data.

In our approach we combine the rationale of these previous approaches. On the one hand, the use of different subspaces, or different projections, is capable of inducing diversity and produces better ensembles. On the other hand, putting more emphasis on difficult instances, as boosting methods do, obtains very good performance and is able to reduce both bias and variance of the classifiers. Nevertheless, boosting is very sensitive to noise and does not usually perform well on small data sets. As the weight of each instance for the training of the classifier depends on whether it is correctly classified too much emphasis may be put on noisy instances or outliers. So, our approach is based on combining the main ideas underlying boosting and random subspace methods, namely:

- All the classifiers should receive all the training instances for learning, and all of them are equally weighted. We are thus neither throwing away any instances which is something which happens in bagging, or putting more emphasis on misclassified instances as in boosting.
- Each classifier uses a different nonlinear projection of the original data onto a space of the same dimension. We are using this to create diversity within the training sets.
- Following the basic principles of boosting, each nonlinear projection is constructed in order to make the classification of difficult instances easier.

This approach is able to incorporate the advantages of boosting without its main drawbacks. The construction of the projection taking into account only instances that have been misclassified by a previous classifier permits the new classifier to focus on difficult instances. Nevertheless, as this classifier receives all the data, the sensitivity to noise and the effect of small data sets is greatly reduced.

In this way, the method presented in this paper is a hybrid of approaches (iii) and (iv) identified by Kuncheva (2001). It uses different feature spaces and only a subset of instances to construct those spaces.

The problem we must solve is how to construct a projection that favours the correct classification of a subset of instances. This problem is solved by means of a neural network as is explained in Section 2.

This paper is organised as follows: Section 2 explains in depth the proposed methodology; Section 3 surveys some related work; Section 4 shows the experimental setup and the results obtained with the proposed algorithm and several standard methods; Section 5 reports some further experiments aimed at understanding the behaviour of the method; and Section 6 states the conclusions of our work.

2. Constructing Nonlinear Projections

The problem of obtaining a useful projection is not trivial. Methods for projecting data are focused on the features of the input space, and do not take into account the labels of the instances. Moreover, most of them are specifically useful for non-labelled data and aimed at data analysis. Among the most widely used of these methods we can cite principal component analysis (PCA) (Jolliffe, 1986), Kohonen's self-organizing maps (Kohonen, 2001), and factor analysis (Gorsuch, 1983). Nevertheless, none of them is appropriate for our problem.

Our approach is based on the use of the projection carried out by the hidden layer of a multilayer perceptron neural network when it is used for classification purposes. In order to get a useful projection, we must take into account the role of the hidden layer in a neural network. As stated in Haykin (1999), p. 199:

> *Hidden neurons* play a critical role in the operation of a multilayer perceptron with back-propagation learning because they act as *feature detectors*. As the learning process progresses, the hidden neurons begin to gradually "discover" the salient features that characterise the training data. They do so by performing a nonlinear transformation on the input data into a new space called the *hidden space*, or *feature space*. In this new space the classes of interest in a pattern-classification task, for example, may be more easily separated from each other than in the original input space.

From this point of view, neural networks can be considered similar to *basis function* models (Denison et al., 2002). These models assume that the function to be implemented, g, is made up of a linear combination of basis functions and corresponding coefficients. Hence g can be written:

$$g(x) = \sum_{i=1}^{H} \beta_i B_i(x), \quad x \in \mathcal{X} \subset \mathbb{R}^D, \tag{1}$$

where $\beta = (\beta_1, \ldots, \beta_k)'$ is the set of coefficients corresponding to basis functions $B = (B_1, \ldots, B_k)$. Typically, the basis functions in (1) are nonlinear transformations. Neural networks can be considered another example of basis function models. Methodologically, there is a major separation in the multilayer perceptron (MLP) approach, as the combination of the basis functions is not always linear, as in (1), and subsequent sets of basis functions, represented by further hidden layers, can be constructed.

Let us consider a feed-forward neural network with D inputs and a hidden layer with H nodes. The hidden layer carries out a non linear projection of input vector x to a vector h where:

$$h_i = f\left(\sum_{j=0}^{D} w_{ij} x_j\right), \qquad x_0 = 1.$$

As we have stated, each node performs a nonlinear projection of the input vector. So, $h = f(x)$, and the output layer obtains its output from vector h. In this context we can consider this projection as a basis function, so $B_i(x) = f\left(\sum_{j=0}^{D} w_{ij} x_j\right)$, and the output of the network is:

$$y(x) = F\left(\sum_{i=1}^{H} \beta_i B_i(x)\right)$$

where F is the transfer function of the output layer, and the β_i represent the weights of the connections from the hidden layer to the output layer. This projection performed by the hidden layer of a multi-layer perceptron distorts the data structure and inter-instance distances (Lerner et al., 1999) in order to achieve a better classification.

So, the projection performed by the hidden layer focuses on making the classification of the instances easier. Our approach to obtain a projection focused on making the classification of difficult instances easier consists of training a neural network with only the instances that have been incorrectly classified by the previous classifier. The number of hidden nodes of the network is the same as the number of input variables, so the hidden layer is performing a nonlinear projection into a space of the same dimension as the input space. As the network has been trained only with a subset of the training instances, the projection performed by the hidden layer focuses on making the classification of only this subset easier.

Once the network is trained, the projection implemented by the hidden layer is used to project all the data set, and these projections are fed to the new classifier to be trained.[1] Then, each classifier performs its task using as input a feature space that is created to make the classification of difficult instances easier.

This nethod can be used with any base classifier as the projection is made before training the classifier and the obtained projected data can be used for any type of classifier. The complexity of the classifiers is not increased, as the feature space into which the data is projected has the same dimension as the original input space.

2.1 Nonlinear Projection Based Algorithm

The initialisation of the network is very important for the overall performance as it has been shown that back-propagation is sensitive to initial conditions (Kolen and Pollack, 1991). For the initialisation of the weights of the networks, we used the method suggested by LeCun et al. (1998) and described in Haykin (1999). The weights are obtained from a uniform distribution within the interval $[-3/\sqrt{D}, 3/\sqrt{D}]$, where D is the number of inputs to the node. In this way, we try to avoid large initial values, that can saturate the transfer function and have a dramatic impact on the performance of the network.

The proposed method for constructing classifier ensembles is shown in Algorithm 1. The proposed algorithm has not incorporated any other ensemble construction method, that may improve its performance, in order not to obscure its behaviour.

Algorithm 1: Nonlinear Boosting Projection algorithm.

```
    Data   : A training set $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, a base learning algorithm, $\mathbb{L}$, and the
             number of iterations $T$.
    Result : The final classifier:
             $C^*(\mathbf{x}) = \arg\max_{y \in Y} \sum_{t:C(\mathbf{x})=y} 1$.
1   $C_0 = \mathbb{L}(S)$
    for $t = 1$ to $T - 1$ do
2       $S' \subset S, S' = \{\mathbf{x}_i \in S : C_{t-1}(\mathbf{x}_i) \neq y_i\}$.
3       Train network $H$ with $S'$ and get projection $\mathbf{P}(\mathbf{x})$ implemented by the hidden layer of $H$.
4       $C_t = \mathbb{L}(\mathbf{P}(S))$
    end
```

We have named the algorithm *Non-Linear "Boosting" Projection* (NLBP) as the projections are constructed using the basic idea of boosting methods. The source code, in C, used for the standard and proposed methods is under GNU License and is freely available upon request to the authors.

Although we need to train an additional neural network for each new classifier of the ensemble, except the first one, the additional complexity of this training process is diminished by the fact that the projection network is trained using only the instances misclassified by the previous classifier.

1. To avoid confusion, we must remark that the network trained using the misclassified instances is not part of the ensemble, and only the projection carried out by its hidden layer is used.

We can illustrate how the method works with a toy example. Consider the simple case of a two-dimensional space and the learning and testing sets shown in Figure 1. The figure shows how there are two points in the training set that are, with a high probability, mislabelled noisy data. Using these two data sets we run our algorithm and ADABOOST with an ensemble of 50 classifiers and a neural network, a C4.5 tree and a support vector machine as base classifiers. The results are shown in Table 1.

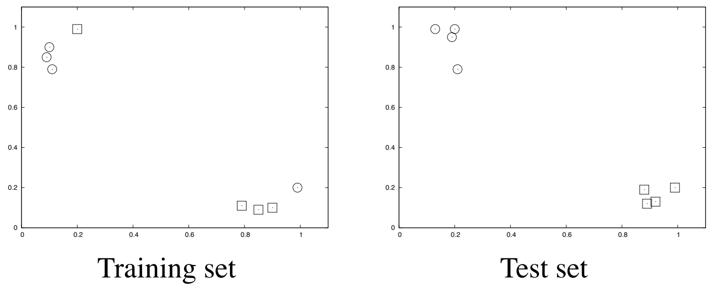


Figure 1: Training and testing data for a two-dimensional problem of two classes with two likely mislabelled noisy instances

The example illustrates the differences of our method with boosting methods and how these differences may constitute an advantage. After the initial classifier is trained, ADABOOST focus its effort on classifying the two misclassified instances correctly. The result is that the learning error drops eventually to 0, but the cost is overlearning the training data and poor generalisation. On the other hand, our method tries to classify the misclassified instances but without allowing the classifier to put too much effort on that task. The result is that the learning error is larger but the generalisation ability is better. This is, of course, a toy problem only intended to illustrate the underlying ideas of our method.

Base classifier	Ensemble method			
	ADABOOST		NLBP	
	Training	Test	Training	Test
Neural network	0.0000	0.5000	0.2500	0.0000
Support vector machine	0.0000	0.2500	0.2500	0.0000
C4.5	0.0000	0.2500	0.1250	0.1250

Table 1: Summary of training and test error results for the toy problem with an ensemble of 50 classifiers.

3. Related Work

The use of different spaces for ensemble construction has been extensive in recent research. Ho (1998) showed that the random subspace method was able to improve the generalisation error. The random subspace method, rooted in the theory of stochastic discrimination (Kleinberg, 2000), has common points with bagging, but instead of sampling instances it samples subspaces (Skurichina and Duin, 2001). The random subspace method has been successfully applied to different problems (Munro et al., 2003; Hall et al., 2003).

Opitz (1999) developed accurate and diverse classifiers for an ensemble using *ensemble feature selection*. In contrast to classical feature selection algorithms, that are focused on finding the best feature subset for a given classifier, ensemble feature selection adds an additional objective of promoting disagreement among the individual classifiers. Different strategies have been proposed to ensemble feature selection, such as hill climbing (Kohavi, 1995; Cunningham and Carney, 2000), a genetic algorithm (Opitz, 1999), and the heuristic method *Ensemble Forward and Backward Sequential Selection* (Aha and Bankert, 1995). A comparative study for medical diagnosis tasks is carried out in Tsymbal et al. (2003).

Utsugi (2001) proposed an ensemble of independent factor analysers (Anderson, 1984). This new statistical model assumes that each data point is generated by the sum of outputs of independently activated factor analysers. A maximum likelihood (ML) estimation algorithm for the parameters is derived using a Monte Carlo EM algorithm with a Gibbs sampler. The independent factor analysers developed into feature detectors that resemble complex cells in mammalian visual systems.

Yand et al. (2000) used mixtures of linear subspaces to create a classifier for face recognition. Rodríguez et al. (2006) proposed a method based on applying PCA to subsets of inputs variables. The method obtained very good results on a large set of real-world problems.

4. Experiments

In order to test the performance of our method on solving classification tasks, we need to compare it with other widely used ensemble methods. In this section we try to make as fair a comparison as possible. So, we have chosen three different base classifiers: a neural network, a C4.5 tree (Quinlan, 1993), and a support vector machine (SVM) (Cristianini and Shawe-Taylor, 2000) . The first two have been widely used for ensemble construction in the literature. The last one has been shown recently to achieve good results as a member of an ensemble (Kim et al., 2002; Diao et al., 2002), although its stability might suggest that it is not a suitable base learner for constructing an ensemble. For the SVM we used a Gaussian kernel and parameters $C = 10.0$ and $\gamma = 0.1$. We used the `LIBSVM` library (Chang and Lin, 2001). For the neural network we used a standard multilayer perceptron trained using a simple back-propagation algorithm (Rumelhart et al., 1986). The network has a hidden layer of 25 nodes and hyperbolic tangent transfer function for the hidden layer and logistic transfer function for the output layer. The network was trained for 100,000 iterations with a learning coefficient $\eta = 0.25$ and a momentum coefficient $\alpha = 0.1$. For C4.5 we used the available code by the author of the algorithm. We must state that these parameters are rather standard and have not been selected in any way to improve the performance of any of the methods.[2]

For all three base classifiers we performed experiments using bagging, Arc-x4, ADABOOST, and MADABOOST (Domingo and Watanabe, 2000).

Before deciding upon bagging we tested its performance against wagging. Wagging is a variant of bagging (Breiman, 1996a) that requires a learning algorithm that can use training instances with different weights. Wagging assigns random weights to the instances of the training set instead of

2. It is obvious that these parameters might not be appropriate for all the data sets. Nevertheless, adjusting the parameters for each data set and each base learner is computationally infeasible. Moreover, as the parameters are common for all the methods, any weakness will be shared for all of them.

resampling the training instances as in bagging. In our implementation the weights assigned to the instances follow a Poisson distribution (Webb, 2000). In a preliminary set of experiments we compared the performance of bagging and wagging, and found that bagging significantly outperformed wagging, so in all the experiments we have used bagging.

We use the Arc-x4 (Breiman, 1998) implementation in Bauer and Kohavi (1999). The factor $(1+e(\mathbf{x})^4)$ that weights each instance is rather heuristic, and the value of 4 for the exponent was experimentally obtained as optimal. These type of methods were named by Breiman (1998) *Arcing* methods after "*A*daptively *r*esampling and *c*ombining".

The ADABOOST algorithm is specifically designed for minimising the exponential loss function:

$$\sum_{i=1}^{m} \exp\left(-y_i f_\lambda(\mathbf{x}_i)\right),$$

where:

$$f_\lambda(\mathbf{x}_i) = \sum_{j=1}^{n} \lambda_j h_j(\mathbf{x}_i).$$

One of the most interesting features of ADABOOST is that the test error continues to improve, even when the learning error reaches 0 (Schapire et al., 1998). However, the marginal error reduction produced by each new classifier tends to decrease. On average, each new classifier has less impact on the test error than the previous members of the ensemble.

We use the ADABOOST version in Webb (2000). For ADABOOST it is required that the *weak learning* algorithm, in our case each individual classifier, achieves an error strictly less than 0.5. This cannot be guaranteed; especially when dealing with difficult multiclass problems. In our experiments when this error is not achieved, we generate a bootstrap sample from the original set and continue the algorithm assigning to that classifier a zero weight (Bauer and Kohavi, 1999; Zhou et al., 2002).

For the three base learners we make a preliminary test to compare resampling and reweighting versions of the boosting algorithms. For neural networks we found that reweighting performed better than resampling; for C4.5 and SVM we found that resampling achieved better results. So, in the experiments reported here for the standard methods we use reweighting for neural networks, and resampling for C4.5 and SVMs.

For the case of a neural network as a base classifier we make some additional experiments to improve the quality of the comparison. Firstly, instead of using standard ADABOOST, we used ADABOOST.MH (Schapire and Singer, 1999, 2000). Moreover, we also used the algorithms GENTLEBOOST (Friedman et al., 2000) and LOGITBOOST (Friedman et al., 2000).[3] Secondly, the hidden weights of the non-linear projection are also learned when using standard methods as an additional layer of the network, in order to avoid any spurious advantage for the proposed method derived from the use of the additional hidden layer represented by the non-linear projection.

All the ensembles are made up of 50 classifiers, regardless of the kind of base classifier. This number is fairly common in the literature. Although desirable, fixing the number of classifiers on

3. These three algorithms modify the learning of the base classifier, so we cannot use them with C4.5 and SVMs as we are using standard libraries for these two base classifiers.

each ensemble taking into account either the problem or the type of base learner is computationally unfeasible. Moreover, it is known (Zenobi and Cunningham, 2001) that the diversity and the accuracy of the ensemble usually plateau at some size between 10 and 50 members. The standard ensembles are also made up of 50 networks. Furthermore, for ADABOOST the error bound of the training error is almost 0 after the addition of about 30 classifiers to the ensemble (Kuncheva, 2003), so 50 is a widely used ensemble size.

4.1 Experimental Setup

For testing the validity of the proposed method we have selected 44 data sets from the UCI Machine Learning Repository (Hettich et al., 1998). A summary of these data sets is shown in Table 2.

Three of the data sets were reduced to make them of a manageable size. For isolet and zip we performed a principal component analysis and retained the first 34 and 50 principal components respectively. For the letter data set we use a randomly selected subset of 5000 instances from the whole set of 20000 instances.

In order to avoid confusion we will use the term “learning error” for referring to the error of a classifier on the training set, and the term “test error” for the error of a classifier on the testing set. In the literature, the terms “generalisation error” and “prediction error” are also common for referring to the error on the test set. However, we believe that these two terms are more appropriate for the probability of misclassifying a new sample, thus the generalisation error is the expected test error.

The experiments were conducted following the 5x2 cross-validation set-up (Dietterich, 1998). We perform five replications of a two-fold cross-validation. In each replication the available data is partitioned into two random equal-sized sets. Each learning algorithm is trained on one set at a time and tested on the other set. The original test proposed by Dietterich suffers from low replicability, so to test the differences between two algorithms we use the 5x2cv F test (Alpaydin, 1999).

This test has the following formulation. We have five replications, $i = 1, \ldots, 5$, and two-folds, $j = 1, 2$, for each replication. $p_i^{(j)}$ is the difference between the error rates of the two classifiers on fold j of replication i. The average on replication i is $\bar{p} = (p_i^{(1)} + p_i^{(2)})/2$, and the estimated variance is $s_i^2 = ((p_i^{(1)} - \bar{p})^2 + ((p_i^{(2)} - \bar{p})^2$. Alpaydin (1999) proposed the test:

$$f = \frac{\sum_{i=1}^{5} \sum_{j=1}^{2} \left(p_i^{(j)}\right)^2}{2 \sum_{i=1}^{5} s_i^2}, \tag{2}$$

that is approximately F distributed with 10 and 5 degrees of freedom. As a general rule we consider a confidence level of 90%. The same 5x2cv partitions were used for all the reported experiments.

In all tables the error measure is $E = \frac{1}{n} \sum_{i=1}^{n} e_i$, where e_i is 1 if instance i is misclassified and 0 otherwise. In the following sections we report all the experiments performed with the proposed method and all the data sets used for testing the method.

The source code, in C, used for the standard and proposed methods as well as the partitions of the data sets, are freely available upon request to the authors.

4.2 Experimental Results

In the first set of experiments we tested our method against the standard ensemble methods. For a neural network as base learner we used bagging, ADABOOST.MH, MADABOOST, LOGITBOOST, GENTLEBOOST, and Arc-x4. Test error results are shown in Table 3. Tables 4 and 5 show the

Data set	Cases	Classes	Features			Inputs
			Continuous	Binary	Nominal	
anneal	898	5	6	14	18	59
audio	226	24	–	61	8	93
autos	205	6	15	4	6	72
balance	625	3	4	–	–	4
breast-cancer	286	2	9	–	–	9
card	690	2	6	4	5	51
dermatology	366	6	1	1	32	34
ecoli	336	8	7	–	–	7
gene	3175	3	–	–	60	120
german	1000	2	6	3	11	61
glass	214	6	9	–	–	9
glass-g2	163	2	9	–	–	9
heart	270	2	6	1	6	13
heart-c	302	2	6	3	4	22
hepatitis	155	2	6	13	–	19
horse	364	3	13	2	5	58
ionosphere	351	2	33	1	–	34
iris	150	3	4	–	–	4
isolet	7797	26	617	–	–	*34*
labor	57	2	8	3	5	29
led24	200	10	–	24	–	24
letter	*5000*	26	16	–	–	16
liver	345	2	6	–	–	6
lrs	531	10	101	–	–	101
lymph	148	4	–	9	6	38
optdigits	5620	10	64	–	–	64
page-blocks	5473	5	10	–	–	10
pendigits	10992	10	16	–	–	16
phoneme	5404	2	5	–	–	5
pima	768	2	8	–	–	8
primary-tumor	339	22	–	14	3	23
promoters	106	2	–	–	57	114
satimage	6435	6	36	–	–	36
segment	2310	7	19	–	–	19
sick	3772	7	2	20	2	33
sonar	208	2	60	–	–	60
soybean	683	19	–	16	19	82
vehicle	846	4	18	–	–	18
vote	435	2	–	16	–	16
vowel	990	11	10	–	–	10
waveform	5000	3	40	–	–	40
yeast	1484	10	8	–	–	8
zip	9298	10	256	–	–	*50*
zoo	101	7	1	15	–	16

Table 2: Summary of data sets. The Inputs column shows the number of inputs to the classifier as it depends not only on the number of input features but also on their type.

results using C4.5 and SVM respectively. As explained, with these two base learners we have used bagging, ADABOOST, MADABOOST, and Arc-x4.

The results in Table 3 show that the proposed method is significantly better than the standard methods for 17 data sets when compared with bagging, 20 with ADABOOST.MH, 19 with GENTLEBOOST, 16 with LOGITBOOST, 17 with MADABOOST and Arc-x4. It is worse than LOGITBOOST in 1 data set and worse than MADABOOST in 2 data sets. The differences are not significant for the other data sets.

Table 4 shows the test error of NLBP and the four standard methods for C4.5 tree as base classifier. As a summary, NLBP is able to significantly improve bagging in 16 data sets, AdaBoost in 13, MadaBoost in 17 and Arc-x4 in 14. It is significantly worse in 3, 5, 3, and 3 data sets respectively. These results show a general advantage of our method. Nevertheless, the experiments using C4.5 as base learner show the worst performance of NLBP. We think the reason is the fact that C4.5 is a tree algorithm that best works with nominal and binary variables; the projection performed by NLBP transform all variables into continuous variables, and that may have a negative effect on C4.5 performance. However, even in this unfavourable scenario NLBP is able to outperform classical methods.

The results in Table 5 show that, for a SVM as base classifier, the proposed method is significantly better than bagging in 19 data sets, than ADABOOST in 23 data sets, than MADABOOST in 16 and than Arc-x4 in 21 data sets. NLBP is significantly worse than the standard methods only once for bagging, MADABOOST, and Arc-x4, and never for ADABOOST. In general terms, the performance of NLBP using a SVM as base classifier, is very good. We believe that the use of a SVM is able to obtain the best of NLBP as the non-linear projections obtained by NLBP are able to make the task of SVM easier.

As a summary, for all the three classifiers NLBP is able to obtain very good results, outperforming both bagging and boosting methods. Of most interest is its performance in some of the most difficult problems, such as breast-cancer, hepatitis, horse, isolet, letter, liver, lymphography, primary-tumor, sonar, and vowel.

5. Analysis of the Proposed Method

In the previous section we have shown that the nonlinear projection approach is very competitive when compared with the standard methods of ensemble construction. In this section we present additional experiments that try to gain some insight into how it works.

In a first set of experiments we investigate if the good behaviour of NLBP is due to any side effect of the proposed algorithm or has its source in the idea of constructing nonlinear projections using only difficult instances. Then, we review the κ-error diagrams and show the κ-error diagrams of the standard and proposed methods. The behaviour of NLBP shown by κ-error diagrams suggests a further study on the sensitivity to noise of the method; that study is performed for all the data sets used in the previous experiments, and the three base classifiers.

5.1 Control Experiments

The first set of experiments of this section is focused on studying whether the basic contribution of our method, that is, using the subset of misclassified instances to obtain a nonlinear projection that makes subsequent classification easier, is responsible for the excellent results reported in the previous section.

Data Set	NLBP	Standard ensemble methods					
		Bagging	AdaB.MH	GentleB	LogitB	MadaB	Arc-x4
anneal	0.0100	0.0114	0.0154	0.0091	0.0512✓	0.0107	0.0103
audiology	0.2557	0.2495	0.2389	0.2531	0.2876	0.2557	0.2584
autos	0.3004	0.3198✓	0.3277✓	0.3257	0.3325	0.3276	0.3296
balance	0.0509	0.0602	0.0522	0.0525	0.0934✓	0.0304✗	0.0330
breast-c	0.3098	0.3168	0.3266	0.3224	0.2951	0.3406✓	0.3056
card	0.1490	0.1498	0.1614	0.1780✓	0.1553	0.1701✓	0.1582
dermatology	0.0246	0.0268	0.0268	0.0295	0.0301	0.0284	0.0268
ecoli	0.1476	0.1494	0.1631✓	0.1923✓	0.1541	0.1583✓	0.1637✓
gene	0.0979	0.1039	0.1026✓	0.1049	0.1017✓	0.1052	0.1015
german	0.2588	0.2620	0.2864✓	0.2802	0.2646	0.2816✓	0.2706✓
glass	0.3206	0.3206	0.3243	0.3327	0.3804✓	0.3271	0.3299
glass-g2	0.2221	0.2257	0.2677✓	0.2478	0.2626✓	0.2551✓	0.2454✓
heart	0.1985	0.2074	0.2096	0.2244✓	0.1941	0.2185✓	0.2215✓
heart-c	0.1762	0.1841	0.1974	0.2027	0.1835	0.2033✓	0.1954
hepatitis	0.1795	0.2094✓	0.2001	0.2064	0.1988	0.2090	0.2051
horse	0.3297	0.3363✓	0.3533	0.3698✓	0.3286	0.3560✓	0.3489✓
ionosphere	0.0980	0.1304✓	0.1418✓	0.1435✓	0.1424✓	0.1418✓	0.1481✓
iris	0.0440	0.0467	0.0493	0.0480	0.0480	0.0493	0.0454
isolet	0.0677	0.0890✓	0.0918✓	0.0806✓	0.1079✓	0.0773✓	0.0886✓
labor	0.1084	0.1964✓	0.1860✓	0.1858✓	0.0981	0.1894✓	0.1930✓
led24	0.3730	0.3780	0.4040✓	0.3910	0.3870	0.3880	0.3900
letter	0.2086	0.2631✓	0.2524✓	0.2194	0.2039✗	0.2075	0.2490✓
liver	0.3084	0.3101	0.3350	0.3523✓	0.3107	0.3246✓	0.3107
lrs	0.1149	0.1085	0.1115	0.1100	0.1247	0.1108	0.1134
lymph	0.1635	0.1878	0.2149✓	0.2135✓	0.1878	0.2189	0.2176✓
optdigits	0.0190	0.0229✓	0.0255✓	0.0197	0.0225✓	0.0218	0.0214✓
page-blocks	0.0328	0.0371✓	0.0379✓	0.0774✓	0.0421	0.0332	0.0351✓
pendigits	0.0104	0.0136✓	0.0142✓	0.0168	0.0381✓	0.0107	0.0131✓
phoneme	0.1936	0.2030	0.1943	0.2133	0.1756	0.1669✗	0.2065
pima	0.2464	0.2500	0.2534	0.2943✓	0.2448	0.2508	0.2451
primary-t	0.5870	0.5940	0.5911✓	0.6253✓	0.5805	0.5975	0.5976
promoters	0.1924	0.2302✓	0.2321✓	0.2302✓	0.1906	0.2302✓	0.2302✓
satimage	0.1061	0.1203✓	0.1216✓	0.1230✓	0.1092	0.1141✓	0.1193✓
segment	0.0332	0.0411✓	0.0398	0.0561✓	0.0594✓	0.0376	0.0454
sick	0.0239	0.0251✓	0.0262✓	0.0557	0.0242✓	0.0241	0.0280
sonar	0.2077	0.2536✓	0.2644✓	0.2683✓	0.2461✓	0.2644✓	0.2673✓
soybean	0.0653	0.0679	0.0682	0.0670	0.1564✓	0.0682	0.0682
vehicle	0.1849	0.1816	0.1983	0.2076	0.1965	0.1790	0.1884
vote	0.0487	0.0552	0.0593	0.0616✓	0.0556	0.0607	0.0616
vowel	0.0731	0.1354✓	0.0820	0.0697	0.2188✓	0.1028✓	0.0731
waveform	0.1373	0.1404	0.1440	0.1580✓	0.1406	0.1434✓	0.1443
yeast	0.4074	0.4123	0.4119	0.4824✓	0.4316	0.4121	0.4189
zip	0.0147	0.0189✓	0.0203✓	0.0169	0.0163✓	0.0175	0.0178✓
zoo	0.0494	0.0631	0.0611	0.0651	0.0812✓	0.0632	0.0651
win/loss		17/0	20/0	19/0	16/1	17/2	17/0

✓/✗ significantly worse/better than NLBP method using F test.

Table 3: Summary of test error results for ensembles using a neural network as base learner and 5x2cv.

Data Set	NLBP	Standard ensemble methods			
		Bagging	AdaBoost	MadaBoost	Arc-x4
anneal	0.0154	0.0165	0.0085	0.0109	0.0109
audiology	0.2672	0.2858✓	0.2434	0.2557	0.2425
autos	0.2907	0.3160	0.2624	0.2750	0.2653
balance	0.0685	0.1527✓	0.2115✓	0.1795✓	0.2125✓
breast-cancer	0.2811	0.2902	0.3245✓	0.3147✓	0.3336✓
card	0.1591	0.1414	0.1449	0.1478	0.1507
dermatology	0.0273	0.0208	0.0323	0.0312	0.0361✓
ecoli	0.1530	0.1750	0.1738✓	0.1756✓	0.1720✓
gene	0.0983	0.0751✗	0.0765✗	0.0718✗	0.0732✗
german	0.2520	0.2534	0.2712✓	0.2626	0.2654✓
glass	0.3150	0.3140	0.3000	0.3028	0.3019
glass-g2	0.2198	0.1756✗	0.1462✗	0.1585	0.1609
heart	0.1859	0.1948	0.2052✓	0.2133	0.2044
heart-c	0.1841	0.1927	0.2046✓	0.2053✓	0.2033
hepatitis	0.1821	0.2066✓	0.2104	0.2118✓	0.2013
horse	0.3258	0.3308	0.3308	0.3258	0.3242
ionosphere	0.0507	0.0815✓	0.0763	0.0798✓	0.0712
iris	0.0467	0.0480	0.0587	0.0573	0.0520
isolet	0.0670	0.2151✓	0.1217✓	0.1545✓	0.1320✓
labor	0.0741	0.1227✓	0.1049	0.1190	0.0978
led24	0.3940	0.3330	0.3680	0.3620	0.3730
letter	0.1128	0.2070✓	0.1196	0.1426✓	0.1243
liver	0.3246	0.3159	0.3252	0.3159	0.3194
lrs	0.1115	0.1454✓	0.1398✓	0.1481✓	0.1428✓
lymphography	0.1838	0.1811	0.1838	0.1973	0.1905
optdigits	0.0193	0.0555✓	0.0206	0.0312✓	0.0242✓
page-blocks	0.0290	0.0285	0.0307	0.0303	0.0298
pendigits	0.0091	0.0271✓	0.0090	0.0186✓	0.0143✓
phoneme	0.1287	0.1378	0.1112✗	0.1137✗	0.1082✗
pima	0.2484	0.2440	0.2612	0.2516	0.2625
primary-tumor	0.5770	0.5705	0.5970	0.5781	0.5858
promoters	0.2189	0.1792	0.2038	0.1793	0.1736
satimage	0.0942	0.1152✓	0.0912	0.0975	0.0931
segment	0.0304	0.0382	0.0233✗	0.0339	0.0280
sick	0.0286	0.0163✗	0.0139✗	0.0156✗	0.0151✗
sonar	0.1836	0.2471✓	0.2462✓	0.2385✓	0.2433✓
soybean	0.0650	0.0764	0.0682	0.0747	0.0679
vehicle	0.2109	0.2650✓	0.2440	0.2610✓	0.2518
vote	0.0506	0.0469	0.0593	0.0620	0.0625
vowel	0.0786	0.1996✓	0.1208✓	0.1556✓	0.1334✓
waveform	0.1387	0.1678✓	0.1625✓	0.1658✓	0.1599✓
yeast	0.4051	0.4063	0.4348✓	0.4116✓	0.4239✓
zip	0.0163	0.0841✓	0.0371✓	0.0523✓	0.0423✓
zoo	0.0848	0.0835	0.0672	0.0653	0.0673
win/loss		16/3	13/5	17/3	14/3

✓/✗ significantly worse/better than NLBP method using F test.

Table 4: Summary of test error results for ensembles using a C4.5 tree as base learner and 5x2cv.

Data Set	NLBP	Standard ensemble methods			
		Bagging	AdaBoost	MadaBoost	Arc-x4
anneal	0.0136	0.0508✓	0.0267	0.0325✓	0.0312✓
audiology	0.2522	0.2893✓	0.2637	0.2540	0.2593
autos	0.2906	0.3179✓	0.3064	0.3063	0.3034
balance	0.0426	0.0691✓	0.0595	0.0624✓	0.0502
breast-cancer	0.2930	0.3091	0.3790✓	0.3448✓	0.3630✓
card	0.1632	0.2162✓	0.2409✓	0.2238✓	0.2377✓
dermatology	0.0246	0.0290	0.0312	0.0301	0.0317
ecoli	0.1512	0.1476	0.1976✓	0.1482	0.1750✓
gene	0.0998	0.1168✓	0.1162✓	0.1207✓	0.1225✓
german	0.2680	0.2992✓	0.2914✓	0.2916✓	0.2946✓
glass	0.3234	0.3383	0.3411	0.3420✓	0.3327
glass-g2	0.2098	0.2248	0.2196	0.2063	0.2210
heart	0.1844	0.1956✓	0.2281✓	0.2119✓	0.2244✓
heart-c	0.1940	0.2126	0.2510✓	0.2331✓	0.2417✓
hepatitis	0.1847	0.1899	0.1948	0.1911	0.1936
horse	0.3440	0.3698✓	0.3692	0.3648✓	0.3654
ionosphere	0.0496	0.0547	0.0644	0.0581	0.0576
iris	0.0467	0.0413	0.0467	0.0413	0.0493
isolet	0.0682	0.1193✓	0.0743	0.0938✓	0.0794✓
labor	0.0875	0.1086	0.1293✓	0.1119	0.1222✓
led24	0.3680	0.3950✓	0.4330✓	0.4090✓	0.4310✓
letter	0.1016	0.1035✓	0.1380✓	0.0940✗	0.1646✓
liver	0.3310	0.3159	0.3623✓	0.3206	0.3333✓
lrs	0.1130	0.1172	0.1221	0.1232	0.1318
lymphography	0.1811	0.2797✓	0.2595✓	0.2568✓	0.2486✓
optdigits	0.0162	0.0180	0.0226✓	0.0200	0.0203✓
page-blocks	0.0347	0.0384✓	0.0401✓	0.0375	0.0382
pendigits	0.0060	0.0057	0.0067	0.0049	0.0053
phoneme	0.1769	0.1565	0.1574	0.1556	0.1585✗
pima	0.2372	0.2526	0.2953✓	0.2505	0.2711✓
primary-tumor	0.5911	0.6365✓	0.6371✓	0.6289✓	0.6318✓
promoters	0.2057	0.1943	0.2094	0.2113	0.2132
satimage	0.0888	0.0904	0.0929	0.0867	0.0933
segment	0.0413	0.0470✓	0.0409	0.0426	0.0408
sick	0.0313	0.0355✓	0.0343✓	0.0339	0.0339✓
sonar	0.1615	0.2414✓	0.2413✓	0.2568✓	0.2615✓
soybean	0.0659	0.0712	0.0732✓	0.0685	0.0752
vehicle	0.1846	0.2213✓	0.2314✓	0.2194✓	0.2234
vote	0.0446	0.0524	0.0570✓	0.0510	0.0538
vowel	0.0467	0.0555	0.0448	0.0418	0.0434
waveform	0.1418	0.1530	0.1697✓	0.1512	0.1624✓
yeast	0.4093	0.4160	0.4291✓	0.4233	0.4443✓
zip	0.0182	0.0158✗	0.0188	0.0170	0.0172
zoo	0.0533	0.0652	0.0455	0.0515	0.0575
win/loss		19/1	23/0	16/1	21/1

✓/✗ significantly worse/better than NLBP method using F test.

Table 5: Summary of test error results ensembles with using a SVM as base learner and 5x2cv.

The first experiment tests if the performance of the method is due to the use of different projections. It is known that the use of random subspaces is able to improve the performance of an ensemble (Ho, 1998), and it is possible that the use of different projections of the original data, no matter how they are obtained, could be the cause of the good performance of NLBP. So we have trained the classifiers that make up the ensemble using a different nonlinear random projection for each classifier. The random nonlinear projection is obtained using the same neural network used in NLBP, but with the weights randomly initialised in the interval $[-1,1]$ and no training.

The second experiment aims to establish if a similar performance can be obtained if we obtain different nonlinear projections using all the instances of the training set instead of only the misclassified instances. Thus, for this simulation, we apply Algorithm 1 but projection $\mathbf{P}(\mathbf{x})$ is constructed using the whole training set, S, instead of the subset of instances incorrectly classified by the previous classifier, S'. The results of these two experiments for the three base classifiers are shown in Table 6.

The table shows how NLBP is performing significantly better than the two control algorithms for many problems. This experiment supports the approach of the proposed algorithm, ruling out as a cause of its good performance the use of an additional layer of learning. For a neural network NLBP is significantly better than a random projection in 16 data sets, and better than the projection using all the instances in 20 data sets. Similar results are obtained for a C4.5 tree and a SVM.

5.2 κ - Error Diagrams

One way of understanding the behaviour of ensemble methods is by means of a κ-error diagram (Margineantu and Dietterich, 1997; Dietterich, 2000a). These diagrams represent a point for each pair of classifiers. The x coordinate is a measure of the diversity of the two classifiers known as κ measure, the y coordinate is the average error of the two classifiers on the test data. The two values are conflicting, as it is obvious that we cannot have both perfect and independent classifiers. The κ-error diagram is the scatter plot of the points corresponding to all pairs of classifiers.

The κ measure is defined as follows: let us consider a problem with K classes, and let $\mathbf{C}$ be a $K \times K$ matrix such that C_{ij} contains the number of instances assigned to class i by the first classifier and to class j by the second classifier. Let us define:

$$\Theta_1 = \frac{\sum_{i=1}^{K} C_{ii}}{n},$$

and

$$\Theta_2 = \sum_{i=1}^{K} \left(\sum_{j=1}^{K} \frac{C_{ij}}{n} \times \sum_{j=1}^{K} \frac{C_{ji}}{n} \right),$$

where n is the number of instances. Then, the κ statistic is defined:

$$\kappa = \frac{\Theta_1 - \Theta_2}{1 - \Theta_2}.$$

When the agreement of the two classifiers equals that expected by chance $\kappa = 0$; when they agree on every instance $\kappa = 1$. Negative values mean a systematic disagreement between the two classifier.

Data Set	Neural network			C4.5			SVM		
	NLBP	Random	All	NLBP	Random	All	NLBP	Random	All
anneal	0.0100	0.0118	0.0096	0.0154	0.0294✓	0.0136	0.0136	0.0243✓	0.0089
audiology	0.2557	0.2734	0.2681	0.2672	0.3823✓	0.2779	0.2522	0.4956✓	0.2487
autos	0.3004	0.3111	0.3394✓	0.2907	0.3404✓	0.2994	0.2906	0.3765	0.2867
balance	0.0509	0.0995✓	0.0851✓	0.0685	0.1002✓	0.0822	0.0426	0.1008✓	0.0877✓
breast-c	0.3098	0.3371	0.3504✓	0.2811	0.2727	0.3140✓	0.2930	0.2769	0.3546✓
card	0.1490	0.1710✓	0.1861✓	0.1591	0.1704	0.1646	0.1632	0.1928✓	0.1785✓
dermatol.	0.0246	0.0263	0.0301	0.0273	0.0306	0.0372✓	0.0246	0.0306	0.0284
ecoli	0.1476	0.1363✓	0.1512✓	0.1530	0.1506	0.1655	0.1512	0.1512	0.1554
gene	0.0979	0.0996	0.1009✓	0.0983	0.2363✓	0.1253✓	0.0998	0.3997✓	0.1008
german	0.2588	0.2626✓	0.2840✓	0.2520	0.2592	0.2750✓	0.2680	0.2686	0.2760
glass	0.3206	0.3383✓	0.3393	0.3150	0.3299✓	0.3421✓	0.3234	0.3524✓	0.3355✓
glass-g2	0.2221	0.2111	0.2417	0.2198	0.2444	0.2259	0.2098	0.2579✓	0.2369✓
heart	0.1985	0.2007	0.2267✓	0.1859	0.1748	0.2037	0.1844	0.1756	0.2348✓
heart-c	0.1762	0.2119	0.2245✓	0.1841	0.1888	0.1927	0.1940	0.2033	0.2199✓
hepatitis	0.1795	0.1949	0.1987	0.1821	0.1820	0.2130	0.1847	0.1860	0.2013
horse	0.3297	0.3346	0.3632✓	0.3258	0.3412	0.3302	0.3440	0.3527	0.3539
ionosph.	0.0980	0.1077✓	0.1521✓	0.0507	0.0524	0.1247✓	0.0496	0.0518	0.1225✓
iris	0.0440	0.0427	0.0480	0.0467	0.0613✓	0.0627✓	0.0467	0.0453	0.0467
isolet	0.0677	0.1038✓	0.0715	0.0670	0.2279✓	0.0875✓	0.0682	0.0734	0.0607✓
labor	0.1084	0.1223	0.1823✓	0.0741	0.1127	0.1543	0.0875	0.0915	0.1575✓
led24	0.3730	0.4000	0.3970✓	0.3940	0.4990✓	0.3790	0.3680	0.4280✓	0.3670
letter	0.2086	0.2692✓	0.2328✓	0.1128	0.2172✓	0.1665✓	0.1016	0.1094	0.1136✓
liver	0.3084	0.3066	0.3344	0.3246	0.3617	0.3554	0.3310	0.3438	0.3414
lrs	0.1149	0.1206	0.1206	0.1115	0.1341✓	0.1255✓	0.1130	0.1729✓	0.1160
lymph.	0.1635	0.1730	0.2108✓	0.1838	0.1824	0.2054	0.1811	0.1865	0.2081✓
optdigits	0.0190	0.0230✓	0.0158	0.0193	0.0494✓	0.0267✓	0.0162	0.0179	0.0142
page-b	0.0328	0.0393✓	0.0340	0.0290	0.0327✓	0.0304	0.0347	0.0373	0.0329
pendigits	0.0104	0.0162✓	0.0137✓	0.0091	0.0133✓	0.0132✓	0.0060	0.0062	0.0075
phoneme	0.1936	0.2103	0.1707	0.1287	0.1554✓	0.1552✓	0.1769	0.2122✓	0.1702
pima	0.2464	0.2396	0.2615	0.2484	0.2458	0.2518	0.2372	0.2354	0.2565✓
primary-t	0.5870	0.5846	0.5905	0.5770	0.5917	0.5793	0.5911	0.5823	0.5852
promoters	0.1924	0.2208	0.2283	0.2189	0.2434	0.2698✓	0.2057	0.5283✓	0.2245✓
satimage	0.1061	0.1154✓	0.1142✓	0.0942	0.1028	0.1032✓	0.0888	0.0891	0.0975✓
segment	0.0332	0.0570✓	0.0380✓	0.0304	0.0459✓	0.0311	0.0413	0.0609✓	0.0369
sick	0.0239	0.0290✓	0.0254	0.0286	0.0373✓	0.0244✗	0.0313	0.0320	0.0251✗
sonar	0.2077	0.2154	0.2558✓	0.1836	0.1789	0.2654✓	0.1615	0.1558	0.2577✓
soybean	0.0653	0.0641	0.0694	0.0650	0.0735	0.0662	0.0659	0.0861✓	0.0682
vehicle	0.1849	0.1936	0.1835	0.2109	0.2624✓	0.2059	0.1846	0.2196✓	0.1870
vote	0.0487	0.0446	0.0570	0.0506	0.0510	0.0556	0.0446	0.0478	0.0570
vowel	0.0731	0.1669✓	0.1810✓	0.0786	0.0845	0.1069✓	0.0467	0.1501✓	0.1729✓
waveform	0.1373	0.1395	0.1407	0.1387	0.1604✓	0.1451	0.1418	0.1550	0.1434
yeast	0.4074	0.4159	0.4119	0.4051	0.4074	0.4171✓	0.4093	0.4102	0.4120
zip	0.0147	0.0215✓	0.0148	0.0163	0.0818✓	0.0212✓	0.0182	0.0268✓	0.0128✗
zoo	0.0494	0.0474	0.0611	0.0848	0.1027	0.0789	0.0533	0.0713✓	0.0533
win/loss		16/0	20/0		20/0	19/1		17/0	17/2

✓/✗ significantly worse/better than NLBP method using F test.

Table 6: Control experiment for NLBP. Test error using a random non-linear projection, and a projection trained using all the instances.

Figures 2, 3, and 4 show κ-error diagrams of the first partition for several data sets with four standard methods and NLBP, and neural networks, C4.5, and SVMs as base classifiers, respectively. These diagrams are fairly representative of the diagrams of all the data sets.

For all the three base classifiers we verify the usual behaviour of bagging and boosting methods. Bagging provides diversity, but to a lesser degree than boosting. On the other hand, boosting's improvement of diversity has the side effect of deteriorating accuracy. NLBP behaviour is midway between these two methods. It is able to improve diversity, but to a lesser degree than boosting, without damaging accuracy as much as boosting. This behaviour suggests that the performance of NLBP in noisy problems can be better than the performance of boosting methods. The next section is devoted to studying the sensitivity to noise of NLBP, and tests that hypothesis.

5.3 Effect of Noise

Several researchers have reported that boosting methods, among them ADABOOST, degrade their performance in the presence of noise. Dietterich (2000a) tested this effect introducing artificial noise in the class labels of different data sets and confirmed this behaviour. In this section we study the sensitivity of our method to noise.

To add noise to the class labels we follow the method of Dietterich (2000a). To add classification noise at a rate r, we chose a fraction r of the instances and changed their class labels to be incorrect choosing uniformly from the set of incorrect labels. We chose all the data sets and three rates of noise, 5%, 10%, and 20%. With this three levels of noise we have performed the experiments using the 5x2cv setup and NLBP, bagging and ADABOOST ensemble methods and compared the results as the level of noise increases.

For the noise study we have used bagging and ADABOOST as the representative of the boosting methods (for neural networks the used method is ADABOOST.MH as in the previous experiments). Tables 7, 8, and 9 show the comparison of the three methods at noise levels 5%, 10%, and 20%, for a neural network, a C4.5 tree and a SVM respectively. First of all, we can corroborate that tables confirm that bagging is less affected by noise than boosting, as has been shown in several papers, for example, (Dietterich, 2000a).

Table 7 shows the results for a neural network. For a noise level of 5% the results are not much different from the results without noise for all the three methods. Bagging performs slightly worse and ADABOOST.MH slightly better. It seems that for this low noise level the algorithms perform as in the case without noise. For 10% of noise the performance of bagging improves, achieving a win/loss record of 13/3 with NLBP, better than the record without noise, 17/0. On the other hand, the performance of ADABOOST.MH drops to a win/loss record of 28/0 with NLBP. Here the sensibility to noise of ADABOOST.MH is clearly evident.

The behaviour with a noise level of 20% is less clear. Bagging's performance drops to a record of 22/1 and ADABOOST.MH improves to 20/2. Two reasons can account for this behaviour: i) all the algorithms are performing badly due to the large amount of noise, so the results have a higher dependency on random factors, ii) the differences in the test error observed on the different partitions can be very high, making the F test, that depends on the variance between the error within the same partition (see Equation 2), less able to conclude that the differences are significant.

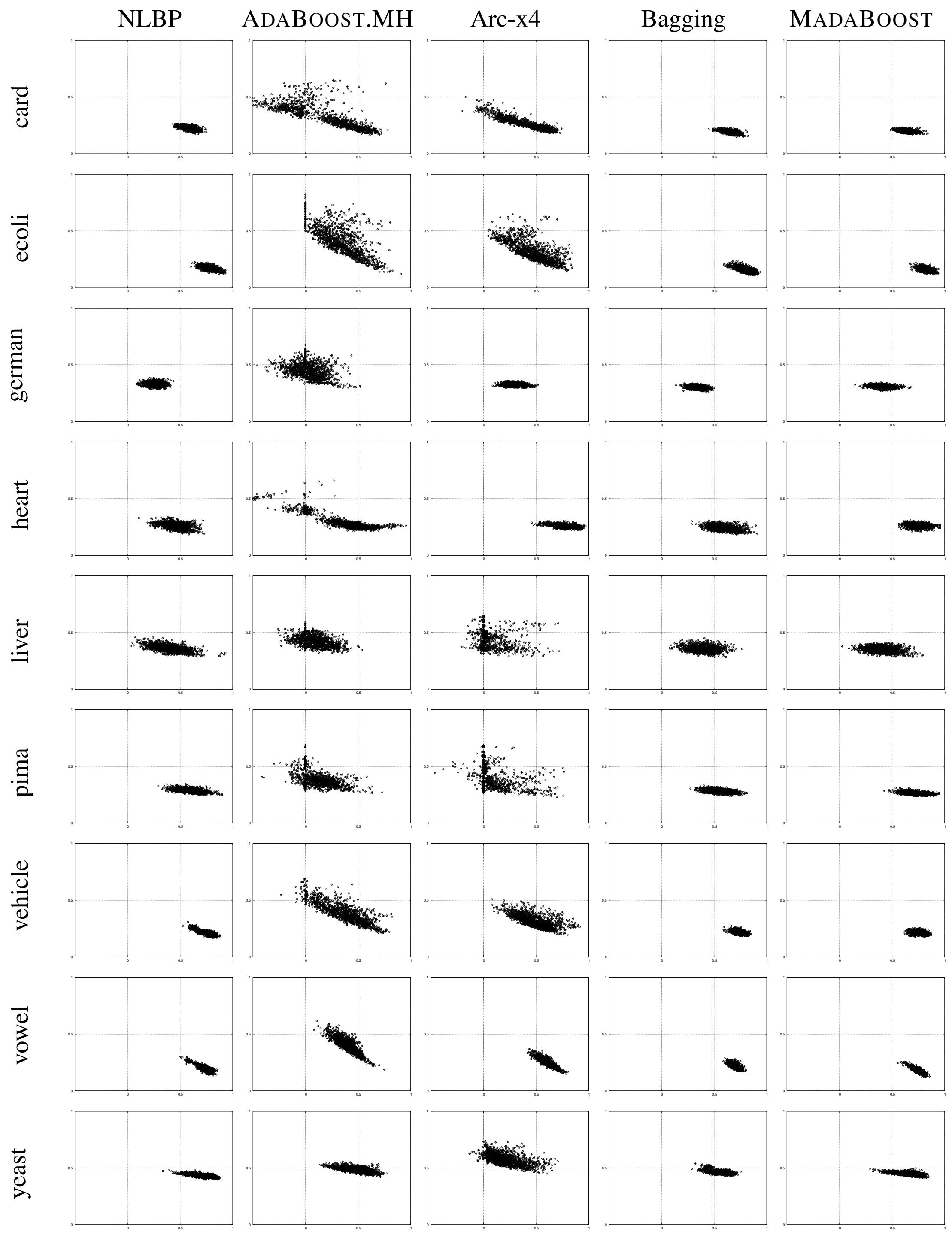


Figure 2: κ-error diagram for the data sets using the five ensemble methods and a neural network as base learner. κ measure is represented on horizontal axis in the interval $[-0.5, 1]$, and error value on vertical axis in the interval $[0, 1]$.

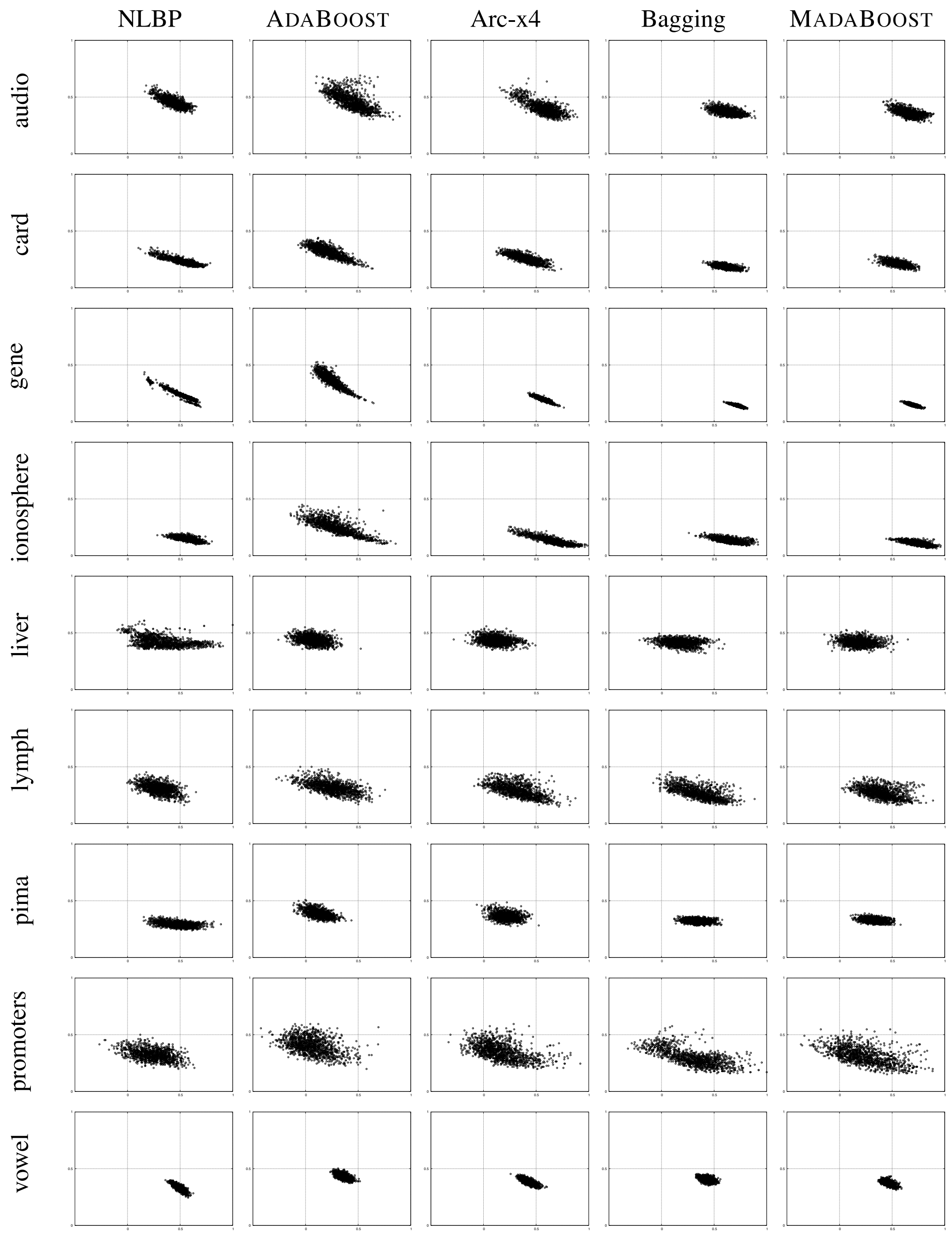


Figure 3: κ-error diagram for the data sets using the five ensemble methods and a C4.5 tree as base learner. κ measure is represented on horizontal axis in the interval $[-0.5, 1]$, and error value on vertical axis in the interval $[0, 1]$.

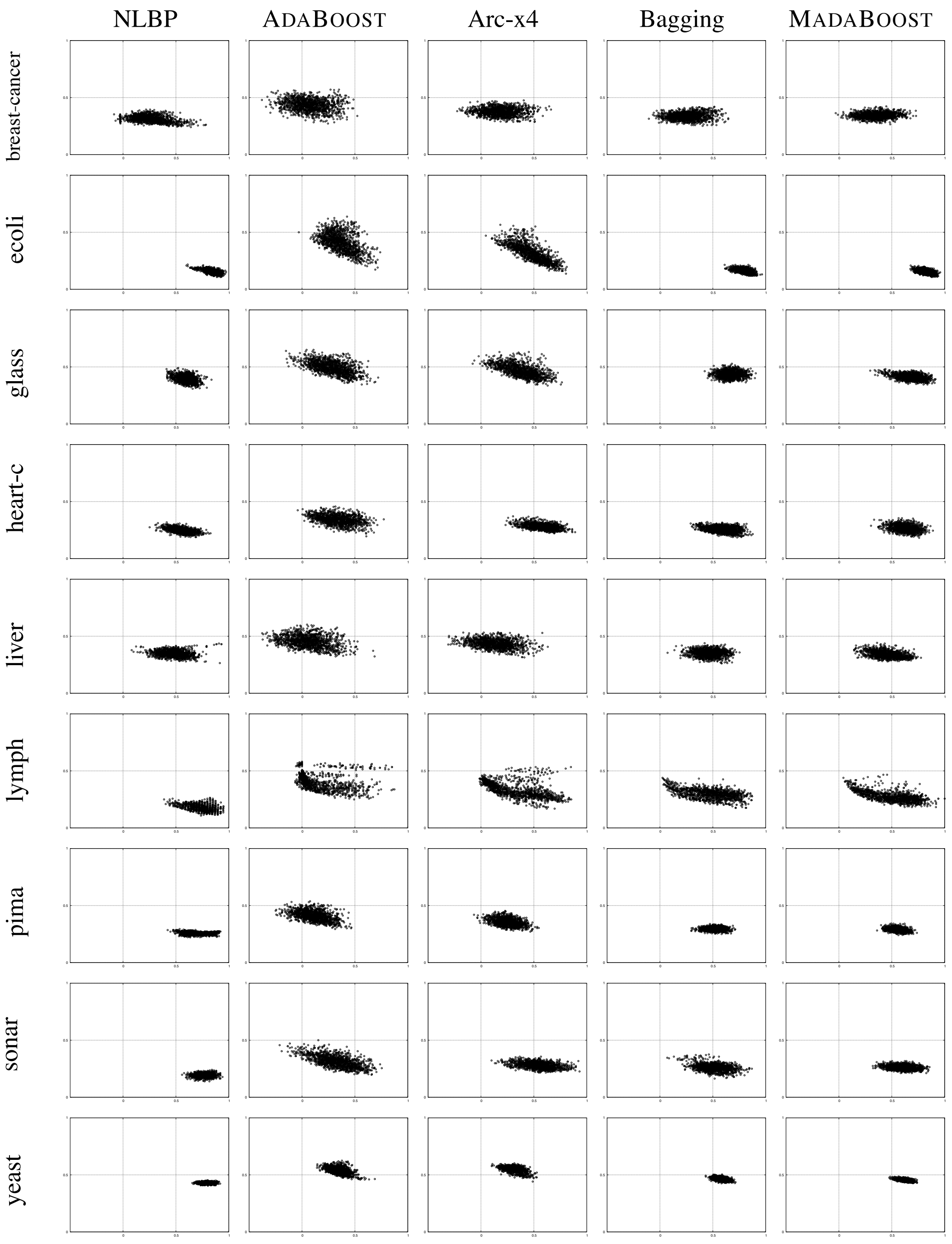


Figure 4: κ-error diagram for the data sets using the five ensemble methods and a SVM as base learner. κ measure is represented on horizontal axis in the interval $[-0.5, 1]$, and error value on vertical axis in the interval $[0, 1]$.

Using C4.5 as base learner, Table 8, the results are much the same. For a noise level of 5% the performance of the three algorithms is similar to the original one. When the noise level increases to 10% the performance of ADABOOST degrades from a win/loss record of 13/5 (with no artificial noise) to a record of 19/2. Bagging slightly worsens compared with NLBP, from a record of 16/3 to 20/6. When the noise level is 20% the effect is more marked for both algorithms, with a record of 18/3 for bagging and 25/1 for ADABOOST.

Table 9 shows a similar behaviour for the three algorithms when using a SVM but with some differences. The performance of ADABOOST dramatically decreases with noise as in the previous cases. For a SVM the negative effect of noise on ADABOOST is even more marked. But the case of bagging is slightly different from the two previous classifiers. With a SVM bagging is less affected by noise, even improving its compared performance with NLBP; for a noise level of 20% bagging is worse than NLBP in only 11 data sets.

In summary, NLBP has the very desirable property of behaving, at least, as robustly as bagging in the presence of noise. This is probably due to the fact that NLBP does not put so much emphasis on incorrectly classified instances as boosting does. However, NLBP does use the incorrectly classified instances in determining the nonlinear representation of the input data which provides it with a performance enhancement over bagging.

5.4 Effect of Ensemble Size

As we have stated, most previous work agrees that boosting methods are fairly resistant to overfitting. Additionally, there is also a general agreement that the most important gain of boosting methods is given by the first few classifiers. These two arguments together support the use of an ensemble of fixed size of 50 base classifiers as reasonable, and that it is not likely that the size of the ensemble might contaminate the experimental results. Nevertheless, it is possible that for some of the problems the ensemble may be either overfitting or underfitting the data. So, we have performed an additional experiment where the size of the ensemble is obtained by a cross-validation strategy. The experiment is carried out using ADABOOST and the proposed NLBP method.

The method for selecting the number of classifiers for each problem is the cross-validation strategy presented in Zhang and Yu (2005). For each partition of each problem we perform a 5-fold cross-validation to obtain the optimal size of the ensemble within the range of $[10, 100]$ classifiers. That is, we divide the training set into five partitions and train the ensemble with four partitions and test the error with the remaining one. This is repeated for each one of the five partitions, and the ensemble size is obtained as the average size of the 5 runs. Then, the algorithm is run with this optimal size of the ensemble and using the whole training set. As the size of the ensemble is estimated for each partition, overfitting or underfitting is less likely to happen. Table 10 shows the results using the three base classifiers of the previous experiments. The parameters of the experiments are the same as the previous runs.

The first noticeable result shown in the table is that the errors of the algorithms using this method are similar to the errors for ensembles of 50 classifiers. These results support the general belief that boosting is quite resistant to overfitting. The results also show that NLBP is also hardly affected by overfitting. The differences among NLBP and ADABOOST are similar to the obtained using 50 classifiers. The significant win/loss record, 21/2, 12/4, and 20/1 for neural network, C4.5 and SVM as base classifiers respectively, is similar to the previous results using 50 classifiers: 20/0, 13/5, and

Data Set	5%			10%			20%		
	NLBP	Bagging	Ada.MH	NLBP	Bagging	Ada.MH	NLBP	Bagging	Ada.MH
anneal	0.0764	0.0909	0.0679	0.1499	0.1254✗	0.1664	0.2904	0.2490✗	0.2608✗
audiology	0.3372	0.3186	0.3425	0.4027	0.4053	0.4372✓	0.5566	0.5655	0.5796✓
autos	0.3851	0.4145✓	0.3975✓	0.4388	0.4278	0.4535	0.5639	0.5834✓	0.5785
balance	0.1030	0.1082	0.1018	0.1530	0.1462	0.1923✓	0.2954	0.2874	0.2928
breast-c	0.3301	0.3664✓	0.3699✓	0.3993	0.3993	0.4413✓	0.4329	0.4273	0.4322
card	0.2148	0.2261	0.2128	0.2458	0.2299	0.2661✓	0.3762	0.3588	0.3632✗
dermatol.	0.0989	0.1180✓	0.1191	0.1628	0.1809✓	0.2071✓	0.2787	0.2874	0.3279✓
ecoli	0.1899	0.2554✓	0.2196	0.2554	0.2643	0.3327✓	0.2994	0.3042	0.3351✓
gene	0.1844	0.1631✗	0.1608✗	0.2079	0.2020	0.2328✓	0.3305	0.3273	0.3395
german	0.2836	0.3142✓	0.3170✓	0.3156	0.3144	0.3436	0.3842	0.3876	0.4096✓
glass	0.3804	0.4037✓	0.3823	0.4561	0.4458	0.4925✓	0.4804	0.4953	0.5168✓
glass-g2	0.2564	0.2761	0.2699✓	0.3375	0.3349	0.3769✓	0.3242	0.3450	0.3254
heart	0.2600	0.2748✓	0.2674	0.3407	0.3185	0.3829✓	0.3741	0.3711	0.3689
heart-c	0.2497	0.2742✓	0.2536	0.3199	0.3046	0.3351	0.4040	0.3815	0.3947
hepatitis	0.2232	0.2555✓	0.2542✓	0.2943	0.2957	0.3345✓	0.3200	0.3355	0.3575✓
horse	0.3632	0.3918	0.3873	0.3973	0.4247✓	0.4297	0.4764	0.4918	0.5165✓
ionosph.	0.1630	0.2268✓	0.1857	0.2376	0.2194	0.2877✓	0.3271	0.2986	0.3248
iris	0.0787	0.1040	0.0973	0.1173	0.1173	0.1773✓	0.3080	0.3360✓	0.3587✓
isolet	0.1107	0.1361✓	0.1448✓	0.1665	0.1912✓	0.1893✓	0.2671	0.3250✓	0.3094✓
labor	0.0912	0.1541✓	0.1860✓	0.2355	0.3054✓	0.2984	0.4457	0.4739✓	0.4739✓
led24	0.4010	0.4440✓	0.4390✓	0.5110	0.5060	0.5240	0.5870	0.6020	0.6120✓
letter	0.1839	0.2394✓	0.3168✓	0.3045	0.3801✓	0.3230✓	0.4038	0.4936✓	0.4922✓
liver	0.3501	0.3559	0.3414	0.3565	0.3443	0.3663	0.4249	0.4285	0.4354
lrs	0.1755	0.1869	0.1676	0.2140	0.2027	0.2249	0.3420	0.3529	0.3405
lymph.	0.2933	0.3378✓	0.3473✓	0.2933	0.3400✓	0.3716✓	0.4041	0.4554✓	0.4581✓
optdigits	0.0711	0.0821✓	0.0755✓	0.1212	0.1252	0.1442✓	0.2208	0.2673✓	0.2304✓
page-b	0.0855	0.0842	0.0897✓	0.1306	0.1410✓	0.1334✓	0.2335	0.2462✓	0.2424
pendigits	0.0625	0.0654✓	0.0660✓	0.1135	0.1165✓	0.1151✓	0.2171	0.2488✓	0.2219✓
phoneme	0.2259	0.2440✓	0.2335✓	0.2588	0.2748✓	0.2558	0.3235	0.3561✓	0.3409✓
pima	0.2719	0.2867	0.2899	0.3073	0.3089	0.3412✓	0.3849	0.3849	0.3896
primary-t	0.6011	0.6212	0.6141	0.6513	0.6595	0.6708	0.7096	0.7356✓	0.7203
promoters	0.2340	0.2472	0.2547	0.2547	0.2962	0.2887	0.4245	0.4472	0.4453
satimage	0.1521	0.1674✓	0.1692✓	0.1988	0.2106✓	0.2128✓	0.3009	0.3154✓	0.3058✓
segment	0.0811	0.0905✓	0.0876✓	0.1322	0.1389	0.1431✓	0.2390	0.2704✓	0.2372
sick	0.0761	0.0792✓	0.0745	0.1297	0.1257✗	0.1418✓	0.2313	0.2402✓	0.2294
sonar	0.2433	0.3106	0.3558✓	0.2914	0.3548✓	0.3414✓	0.3606	0.4010✓	0.4125✓
soybean	0.1318	0.1488✓	0.1280	0.1833	0.1719	0.1921	0.3101	0.3330✓	0.2977
vehicle	0.2525	0.2648✓	0.2624	0.2799	0.2730	0.3109✓	0.3801	0.3965✓	0.3868
vote	0.1195	0.1311	0.1140	0.1710	0.1766	0.2014	0.2427	0.2405	0.2519
vowel	0.2039	0.1756✗	0.2146	0.2499	0.2572	0.2731	0.3432	0.4390✓	0.3701✓
waveform	0.1808	0.1867	0.1822	0.2250	0.2233✗	0.2475✓	0.3102	0.3208✓	0.3115
yeast	0.4412	0.4522	0.4473	0.4747	0.4771	0.4699	0.5220	0.5448✓	0.5283
zip	0.0668	0.0769✓	0.0698✓	0.1207	0.1220✓	0.1295✓	0.2142	0.2374✓	0.2199✓
zoo	0.1760	0.2278	0.1900	0.1642	0.1840✓	0.1977✓	0.2535	0.2813✓	0.2773
win/loss		24/2	18/1		13/3	28/0		22/1	20/2

✓/✗ significantly worse/better than NLBP method using F test.

Table 7: Summary of test error results for ensembles with a neural network using 5x2cv with noise levels of 5%, 10%, and 20%.

Data Set	5%			10%			20%		
	NLBP	Bagging	AdaB	NLBP	Bagging	AdaB	NLBP	Bagging	AdaB
anneal	0.0722	0.0973✓	0.0922	0.1399	0.1817✓	0.1757✓	0.2702	0.3156✓	0.3200✓
audiology	0.3460	0.3027	0.3106	0.3717	0.3566✗	0.3593✗	0.5620	0.5195✗	0.5204✗
autos	0.3871	0.3413✗	0.3725	0.4214	0.3853	0.3911	0.5288	0.5307	0.5367
balance	0.1296	0.3567✓	0.2586✓	0.1718	0.2883✓	0.3082✓	0.3139	0.4198✓	0.4292✓
breast-c	0.3014	0.3007	0.3469	0.3455	0.3881✓	0.4028✓	0.3993	0.4182	0.4392✓
card	0.2029	0.1893	0.1942	0.2371	0.2313	0.2386	0.3495	0.3551	0.3649
dermatol.	0.0891	0.1060	0.0951	0.1563	0.1836✓	0.1776✓	0.2311	0.2475✓	0.2503✓
ecoli	0.1976	0.2125✓	0.2167	0.2494	0.2923✓	0.2976✓	0.2881	0.3298✓	0.3500✓
gene	0.1542	0.1312✗	0.1451	0.2023	0.1746✗	0.1996	0.3242	0.2905✗	0.3227
german	0.2916	0.2912	0.2974	0.3126	0.3320	0.3292	0.3796	0.3904	0.4086✓
glass	0.3841	0.3710	0.3729	0.4393	0.4047	0.4112	0.4738	0.4467	0.4645
glass-g2	0.2627	0.2418	0.2455	0.3350	0.2982	0.3153	0.3316	0.3168	0.3608
heart	0.2445	0.2607	0.2652	0.2941	0.3200	0.3437✓	0.3378	0.3667	0.3830
heart-c	0.2311	0.2583✓	0.2583✓	0.2921	0.3066	0.3175✓	0.3536	0.3960✓	0.3987
hepatitis	0.2259	0.2296	0.2308	0.3021	0.2763✗	0.2983✗	0.2956	0.3047	0.3073
horse	0.3637	0.3522	0.3610	0.3923	0.3995	0.4099	0.4698	0.5066	0.5094✓
ionosph.	0.1459	0.1305	0.1464	0.1715	0.1881	0.1898	0.2706	0.2957	0.3031
iris	0.1027	0.1094	0.1027	0.1147	0.1707✓	0.1720✓	0.3173	0.3560✓	0.3667✓
isolet	0.1152	0.2102✓	0.1706✓	0.1646	0.2489✓	0.2189✓	0.2635	0.3215✓	0.3202✓
labor	0.1050	0.1085	0.1117	0.1863	0.1898	0.1792	0.3576	0.3825	0.3785
led24	0.4350	0.4120	0.4150	0.5270	0.4850	0.4970	0.5860	0.5480	0.5610
letter	0.1645	0.2106✓	0.1893✓	0.2137	0.2499✓	0.2502✓	0.3222	0.3709✓	0.3782✓
liver	0.3768	0.3698	0.3565	0.3843	0.3663	0.3722	0.4493	0.4337	0.4482
lrs	0.1737	0.1936	0.1959✓	0.2098	0.2219✓	0.2174	0.3081	0.3326	0.3296
lymph.	0.2838	0.2838	0.2865	0.2906	0.3203	0.3230	0.3460	0.3851✓	0.4081✓
optdigits	0.0700	0.0873✓	0.0716	0.1207	0.1352✓	0.1244	0.2223	0.2310✓	0.2311✓
page-b	0.0794	0.0842✓	0.0849✓	0.1311	0.1400✓	0.1389✓	0.2312	0.2528✓	0.2526✓
pendigits	0.0588	0.0680✓	0.0614✓	0.1080	0.1167✓	0.1154✓	0.2118	0.2247✓	0.2245✓
phoneme	0.2110	0.1814	0.1716✗	0.2563	0.2075✗	0.2245	0.3247	0.3007✗	0.3340
pima	0.2794	0.2857	0.2927	0.3169	0.3372	0.3331	0.3818	0.3974	0.4083✓
primary-t	0.5941	0.6135	0.6064	0.6430	0.6519	0.6814	0.6902	0.7126	0.7338✓
promoters	0.2358	0.2113	0.1887	0.2793	0.2264✗	0.2547	0.4151	0.4170	0.4170
satimage	0.1385	0.1476✓	0.1391	0.1833	0.1903✓	0.1860✓	0.2849	0.2853	0.2864
segment	0.0820	0.0827	0.0887✓	0.1345	0.1448✓	0.1483✓	0.2446	0.2549	0.2613✓
sick	0.0871	0.0674✗	0.0720✗	0.1422	0.1290✗	0.1442	0.2387	0.2521	0.2818✓
sonar	0.3039	0.2692	0.2769	0.3433	0.3308	0.3260	0.3442	0.3577	0.3837✓
soybean	0.1256	0.1616✓	0.1529	0.1754	0.2234✓	0.2100✓	0.3051	0.3640✓	0.3643✓
vehicle	0.2645	0.3024✓	0.3111✓	0.2976	0.3303✓	0.3343✓	0.4047	0.4388✓	0.4310
vote	0.1154	0.1310	0.1297	0.1697	0.1904✓	0.1959	0.2349	0.2630✓	0.2731✓
vowel	0.1758	0.2085✓	0.1957	0.2244	0.2628✓	0.2554✓	0.3321	0.3667✓	0.3675✓
waveform	0.1821	0.2028✓	0.2045✓	0.2260	0.2432✓	0.2487✓	0.3142	0.3292	0.3364✓
yeast	0.4464	0.4580	0.4625✓	0.4791	0.4945	0.4988	0.5367	0.5566✓	0.5616✓
zip	0.0676	0.1107✓	0.0874✓	0.1201	0.1596✓	0.1404✓	0.2171	0.2390✓	0.2383✓
zoo	0.1859	0.1799	0.2016	0.1841	0.1702	0.2038	0.2653	0.2594	0.2713
win/loss		15/3	12/2		20/6	19/2		18/3	25/1

✓/✗ significantly worse/better than NLBP method using F test.

Table 8: Summary of test error results for ensembles with a C4.5 tree using 5x2cv with noise levels of 5%, 10%, and 20%.

Data Set	5%			10%			20%		
	NLBP	Bagging	AdaB	NLBP	Bagging	AdaB	NLBP	Bagging	AdaB
anneal	0.0757	0.1031✓	0.1238✓	0.1635	0.1682	0.2007✓	0.3114	0.3027	0.3575✓
audiology	0.3469	0.3522	0.5611✓	0.5620	0.4265✗	0.4310✗	0.6691	0.5761✗	0.6035
autos	0.3881	0.3803	0.4671✓	0.4643	0.4262	0.4418	0.5630	0.5463	0.5727
balance	0.1088	0.1264✓	0.1760✓	0.1626	0.1680	0.2003✓	0.3094	0.3146	0.3421✓
breast-c	0.3182	0.3602✓	0.4147✓	0.3727	0.3930✓	0.4301✓	0.4154	0.4273	0.4594
card	0.2180	0.2658✓	0.3049✓	0.2826	0.2907	0.3249✓	0.3875	0.3899	0.4160✓
dermatol.	0.1011	0.0896	0.1465✓	0.1667	0.1656	0.2197✓	0.2574	0.2579	0.3344✓
ecoli	0.1887	0.1839	0.2732✓	0.2476	0.2488	0.3226✓	0.2792	0.2798	0.3607✓
gene	0.1857	0.2050✓	0.3099✓	0.3482	0.2584	0.2869	0.3244	0.3707✓	0.4001✓
german	0.3000	0.3150✓	0.3150	0.3242	0.3310✓	0.3394	0.3780	0.3800	0.3872
glass	0.3729	0.3841	0.4028	0.4327	0.4439	0.4823	0.4682	0.4907✓	0.5252✓
glass-g2	0.2651	0.2652	0.2565	0.3276	0.3117✗	0.3351	0.3192	0.3266	0.3425
heart	0.2333	0.2459	0.2852✓	0.2830	0.3104✓	0.3622✓	0.3452	0.3763	0.3978✓
heart-c	0.2537	0.2596✓	0.3192✓	0.2914	0.3033	0.3576✓	0.3715	0.3854	0.4245✓
hepatitis	0.2194	0.2299	0.2284	0.2866	0.2724	0.2879	0.2866	0.3111	0.3278
horse	0.3709	0.3989✓	0.3989	0.4116	0.4220	0.4160	0.4846	0.4808	0.4879
ionosph.	0.1026	0.1066	0.1391✓	0.1738	0.1721	0.2165✓	0.2684	0.2712	0.3253✓
iris	0.0746	0.0760	0.1013	0.1027	0.1040	0.1414	0.3133	0.3013	0.3347
isolet	0.1101	0.1675✓	0.1721✓	0.1839	0.2144✓	0.2286✓	0.2893	0.3163✓	0.3375✓
labor	0.0979	0.0915	0.1224	0.2005	0.2246	0.2491✓	0.3510	0.3791	0.4179✓
led24	0.4040	0.4270	0.4730✓	0.5270	0.5120	0.5360	0.5910	0.5740	0.6010
letter	0.1859	0.1716✗	0.2432	0.2399	0.2286	0.2683✓	0.3507	0.3647✓	0.4168✓
liver	0.3519	0.3478	0.3762	0.3629	0.3588	0.3890	0.4209	0.4238	0.4592✓
lrs	0.1752	0.1789	0.2079✓	0.2554	0.2268	0.2931	0.3627	0.3439	0.4620
lymph.	0.2838	0.3392✓	0.3176	0.2906	0.4054✓	0.3325	0.3730	0.4297✓	0.4216
optdigits	0.0730	0.0975✓	0.1339✓	0.1276	0.1736✓	0.2368✓	0.2453	0.2857✓	0.3411✓
page-b	0.0865	0.0894	0.0919	0.1346	0.1495✓	0.1388	0.2381	0.2598✓	0.2396
pendigits	0.0605	0.0571✗	0.0997✓	0.1113	0.1092	0.1611✓	0.2171	0.2176	0.2375✓
phoneme	0.2155	0.1988✗	0.1990	0.2496	0.2333	0.2362	0.3171	0.3048	0.3086
pima	0.2755	0.2825	0.3253✓	0.3047	0.3250	0.3664✓	0.3750	0.3878	0.4255✓
primary-t	0.6094	0.6542✓	0.6537✓	0.6519	0.6955✓	0.6955✓	0.7209	0.7327	0.7339
promoters	0.2075	0.1981	0.2226	0.2528	0.2604	0.2812	0.4736	0.4208	0.4227
satimage	0.1500	0.1401✗	0.1630✓	0.1969	0.1877✗	0.2244✓	0.2947	0.2930	0.3498✓
segment	0.0915	0.0994✓	0.1251✓	0.1547	0.1545	0.1829✓	0.2464	0.2563✓	0.2856✓
sick	0.1037	0.0863✗	0.1127✓	0.1505	0.1439✗	0.1785✓	0.2424	0.2479	0.2969✓
sonar	0.2250	0.2914✓	0.3240✓	0.2798	0.3442✓	0.3442✓	0.3356	0.3846✓	0.4010✓
soybean	0.1707	0.1423	0.1699	0.2264	0.1994	0.2407	0.3719	0.3327✗	0.3857
vehicle	0.2456	0.2896✓	0.3104✓	0.2757	0.3187✓	0.3612✓	0.3913	0.4348✓	0.4875✓
vote	0.1117	0.1196	0.1449✓	0.1628	0.1825	0.1977	0.2290	0.2524	0.2763✓
vowel	0.1394	0.1366	0.1703✓	0.1994	0.2077	0.2642✓	0.3174	0.3123	0.3950✓
waveform	0.1955	0.1899	0.2118✓	0.2379	0.2340	0.2436✓	0.3178	0.3249	0.3295✓
yeast	0.4445	0.4507	0.4688✓	0.4725	0.4873✓	0.4965✓	0.5217	0.5368	0.5040
zip	0.0752	0.0896✓	0.1296✓	0.1305	0.1445✓	0.1801✓	0.2276	0.2418✓	0.2591✓
zoo	0.1701	0.1701	0.1743	0.1603	0.1603	0.1821	0.2358	0.2317	0.3031✓
win/loss		16/5	29/0		12/4	25/1		11/2	27/0

✓/✗ significantly worse/better than NLBP method using F test.

Table 9: Summary of test error results for ensembles with a SVM using 5x2cv with noise levels of 5%, 10%, and 20%.

Data Set	Neural network		C4.5		SVM	
	NLBP	ADABOOST.MH	NLBP	ADABOOST	NLBP	ADABOOST
anneal	0.0100	0.0100	0.0118	0.0102	0.0102	0.0278✓
audiology	0.2513	0.2611	0.2823	0.2504	0.2938	0.2779
autos	0.3082	0.3247✓	0.2867	0.2633	0.2945	0.3121
balance	0.0544	0.0502	0.0803	0.2038✓	0.0570	0.0614
breast-c	0.3336	0.3769✓	0.2762	0.3189✓	0.2944	0.3748✓
card	0.1487	0.1797✓	0.1548	0.1551	0.1603	0.2392✓
dermatol.	0.0273	0.0268	0.0301	0.0317	0.0323	0.0361
ecoli	0.1607	0.2006✓	0.1506	0.1768✓	0.1542	0.1863✓
gene	0.0977	0.1036✓	0.1295	0.0798✗	0.1158	0.1208
german	0.2724	0.2868	0.2668	0.2740	0.2892	0.2968
glass	0.3327	0.3505	0.3252	0.3019	0.3187	0.3393✓
glass-g2	0.2197	0.2466✓	0.2284	0.1841	0.2136	0.2858✓
heart	0.1963	0.2356✓	0.1852	0.2208✓	0.1845	0.2089✓
heart-c	0.1835	0.2238✓	0.1801	0.2172✓	0.1861	0.2043✓
hepatitis	0.2052	0.2156✓	0.1936	0.2001	0.2001	0.2000
horse	0.3412	0.3725✓	0.3297	0.3423	0.3330	0.3643✓
ionosph.	0.1282	0.1492✓	0.0638	0.0792✓	0.0980	0.0667
iris	0.0507	0.0454	0.0427	0.0534	0.0427	0.0493
isolet	0.0715	0.0673	0.0756	0.1258✓	0.0649	0.1358✓
labor	0.1432	0.1541✓	0.1192	0.1087	0.1468	0.2006✓
led24	0.3750	0.4130✓	0.4300	0.3810	0.3850	0.4290✓
letter	0.1734	0.1893	0.1172	0.1236	0.1022	0.1006
liver	0.3217	0.3414✓	0.3339	0.3327	0.3426	0.3518
lrs	0.1119	0.1266✓	0.1179	0.1364✓	0.1205	0.1251
lymph.	0.1757	0.2095✓	0.1757	0.1878	0.1932	0.2662✓
optdigits	0.0204	0.0187	0.0241	0.0222	0.0222	0.0221
page-b	0.0339	0.0340	0.0317	0.0303	0.0333	0.0411✓
pendigits	0.0107	0.0064✗	0.0090	0.0095	0.0059	0.0066
phoneme	0.1844	0.1614	0.1405	0.1096✗	0.1712	0.1546
pima	0.2471	0.2695✓	0.2482	0.2674	0.2375	0.2914✓
primary-t	0.5698	0.6064✓	0.5822	0.5970	0.6141	0.6366
promoters	0.2094	0.2227	0.2264	0.2245	0.2302	0.2189
satimage	0.1067	0.1097✓	0.0955	0.0938	0.0939	0.0965
segment	0.0433	0.0363	0.0382	0.0225✗	0.0371	0.0410
sick	0.0465	0.0270	0.0399	0.0151✗	0.0566	0.0344✗
sonar	0.2231	0.2519✓	0.2202	0.2375	0.2087	0.2452✓
soybean	0.0656	0.0706	0.0679	0.0647	0.0688	0.0726
vehicle	0.2002	0.2017	0.2445	0.2513	0.2185	0.2307
vote	0.0529	0.0620	0.0556	0.0556	0.0483	0.0579
vowel	0.0972	0.0717✗	0.0937	0.1301✓	0.0576	0.0456
waveform	0.1361	0.1493✓	0.1473	0.1685✓	0.1481	0.1639✓
yeast	0.4186	0.4146	0.4054	0.4408✓	0.4151	0.4320✓
zip	0.0153	0.0139	0.0206	0.0380✓	0.0160	0.0188✓
zoo	0.0592	0.0592	0.0789	0.0713	0.0474	0.0903✓
win/loss		21/2		12/4		20/1

✓/✗ significantly worse/better than NLBP method using F test.

Table 10: Summary of test error results for ensembles with a neural network, a C4.5 tree and a SVM as base classifiers and the size of the ensemble obtained by cross-validation.

23/0. We can say that the conclusions obtained using 50 classifiers also hold for the results obtained estimating the size of the ensemble by cross-validation.

Table 11 shows the average size of the ensembles. The most interesting result is the fact that most of the ensembles are in the interval between 20 and 30 classifiers. As we have stated, this agrees with most previous work that found little gain in terms of error after about 25 classifiers had been added. It is also interesting to note that, typically, both methods, NLBP and ADABOOST find similar sizes of the ensemble.

6. Conclusions and Future Work

In this paper we have presented a new approach for ensemble construction based on nonlinear projections of the original training instances. The projections are obtained by means of a neural network and are aimed to make the classification of difficult instances easier. This idea is taken from boosting methods, but in contrast with these methods our method does neither resample or weight the instances in the training set. This avoids the drawbacks of either resampling/reweighting, such as poor performance on small data sets and sensitivity to noise.

The only drawback of the proposed method is the necessity of training an additional neural network each time a new classifier must be added to the ensemble. Nevertheless, this network is only trained with the instances misclassified by the previous classifier, so the time needed for its training is much reduced.

We have compared the performance of this method against the performance of bagging, Arc-x4 and several boosting methods. The comparison was made using a fairly large set of real-world problems. Our method showed very good results in terms of test error that clearly outperformed bagging, Arc-x4, and different boosting algorithms. κ-error diagrams showed the ability of the proposed algorithm to improve diversity among classifiers without dramatically affecting accuracy. This improvement of performance has been assessed using ensembles of fixed size and ensembles of variable size obtained by cross-validation.

Additional experiments have been made to test the sensitivity to noise of this approach. These experiments have shown that NLBP is less sensitive to noise than boosting methods, and at least as good as bagging.

We believe that this work opens a new and interesting approach to ensemble construction. Based on the main idea of boosting, that is putting more effort into difficult instances, our approach tries to make the classification of these instances easier without disregarding the instances correctly classified so far. The advantages of this approach have been highlighted by the results presented in this paper. Moreover, an interesting research line opens in which we can devise alternative projection methods that take into account the classification of a subset of instances.

Our future work is mainly directed towards two goals. Firstly, we are working on developing a more theoretical view of our method, that may explain the good experimental results reported here. Secondly, we have experienced some problems with the use of a neural network for the non-linear projections, due to the facility with which the outputs of the hidden nodes of the network saturate, making the projection less efficient. We are working on alternative supervised non-linear projection that can improve the performance of the method.

Data Set	Neural network		C4.5		SVM	
	NLBP	ADABOOST.MH	NLBP	ADABOOST	NLBP	ADABOOST
anneal	16.9	16.8	20.0	18.9	17.4	19.3
audiology	17.7	19.4	25.5	20.5	18.2	14.1
autos	16.0	20.0	21.1	20.6	19.0	13.8
balance	21.6	20.7	21.6	23.0	23.0	21.5
breast-c	20.4	20.1	21.1	20.8	18.6	20.1
card	23.2	23.6	22.0	24.1	24.9	21.7
dermatol.	16.9	17.5	16.7	19.6	18.1	12.6
ecoli	20.5	20.8	20.5	21.9	19.7	20.0
gene	28.5	23.2	25.0	28.8	29.9	16.5
german	23.3	21.5	19.5	24.2	18.9	21.9
glass	18.5	20.1	21.0	22.8	18.0	22.3
glass-g2	19.8	21.5	20.8	22.1	19.6	15.1
heart	18.6	20.0	22.0	24.8	19.0	13.7
heart-c	21.4	22.1	20.8	22.5	19.0	14.6
hepatitis	18.7	18.9	19.6	21.2	18.3	20.5
horse	20.6	20.5	22.1	22.8	21.1	18.1
ionosph.	20.3	18.0	18.0	22.9	18.5	19.1
iris	15.5	16.2	16.8	17.1	15.5	16.7
isolet	28.5	27.9	46.0	49.2	15.7	11.3
labor	16.0	16.0	15.3	17.5	16.0	16.3
led24	20.0	16.4	24.0	17.9	17.5	12.8
letter	28.5	38.5	41.2	36.7	17.2	22.5
liver	23.3	21.0	18.5	23.5	16.0	22.0
lrs	19.0	18.9	19.2	23.0	19.5	13.7
lymph.	17.1	17.5	20.1	21.4	19.3	21.7
optdigits	24.4	26.2	30.5	35.4	15.9	14.8
page-b	23.4	21.6	23.0	21.9	21.0	20.4
pendigits	26.2	25.6	28.7	34.4	18.7	21.8
phoneme	20.0	25.2	21.5	31.9	17.0	22.3
pima	25.0	21.4	24.7	22.2	22.4	22.1
primary-t	21.0	10.8	13.9	11.3	11.4	10.4
promoters	16.5	17.0	16.0	22.1	14.0	12.8
satimage	23.3	26.4	26.6	33.4	16.4	20.2
segment	23.3	21.6	25.0	24.2	23.8	23.2
sick	15.5	22.4	19.7	23.8	15.0	20.6
sonar	19.5	19.2	22.0	23.6	22.0	14.3
soybean	20.1	20.6	18.0	22.1	20.7	14.6
vehicle	21.5	22.2	22.9	26.7	23.8	22.1
vote	18.4	18.6	17.9	20.5	19.5	19.6
vowel	24.0	25.0	27.9	30.7	22.8	22.1
waveform	23.0	25.5	28.4	33.9	16.8	17.1
yeast	22.6	24.5	29.0	22.0	22.3	21.9
zip	23.9	25.1	39.4	41.9	12.2	15.2
zoo	14.9	15.5	18.5	16.9	15.1	18.1

Table 11: Summary of sizes for ensembles with a neural network, a C4.5 tree and a SVM as base classifiers and the size of the ensemble obtained by cross-validation.

References

D. W. Aha and R. L. Bankert. A comparative evaluation of sequential feature selection algorithms. In D. Fisher and H. Lenz, editors, *Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics*, pages 1–7, 1995.

E. Alpaydin. Combined 5×2 cv F test for comparing supervised classification learning algorithms. *Neural Computation*, 11:1885–1892, 1999.

T. W. Anderson. *An Introduction to Multivariate Statistical Analysis*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, New York, 2nd edition, 1984.

E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1/2):105–142, July/August 1999.

L. Breiman. Stacked regressions. *Machine Learning*, 24(1):49–64, 1996a.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996b.

L. Breiman. Bias, variance, and arcing classifiers. Technical Report 460, Department of Statistics, University of California, Berkeley, CA, 1996c.

L. Breiman. Arcing classifiers. *Annals of Statistics*, 26:801–824, 1998.

L. Breiman. Prediction games and arcing algorithms. *Neural Computation*, 11(7):1493–1517, 1999.

N. H. Bshouty and D. Gavinsky. On boosting with polynomially bounded distributions. *Journal of Machine Learning Research*, 3:483–506, 2002.

Ch-Ch. Chang and Ch-J. Lin. *LIBSVM: A Library for Support Vector Machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/ cjlin/libsvm`.

K. Chen, L. Wang, and H. Chi. Methods of combining multiple classifiers with different features and their applications to text-independent speaker identification. *Journal of Pattern Recognition and Artificial Intelligence*, 11(3):417–445, 1997.

K. Cherkauer. Human expert-level performance on a scientific image analysis task by a system using combined artificial neural networks. In *Working Notes of the AAAI Workshop on Integrating Multiple Learned Models*, pages 15–21, 1996.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, 2000.

P. Cunningham and J. Carney. Diversity versus quality in classification ensembles based on feature selection. In R. L. de Mantarás and E. Plaza, editors, *Proceedings of the Eleventh Conference on Machine Learning ECML 2000*, pages 109–116, Barcelona, Spain, 2000. Springer.

D. G. T. Denison, C. C. Holmes, B. K. Mallick, and A. F. M. Smith. *Bayesian Methods for Nonlinear Classification and Regression*. Wiley Series in Probability and Statistics. John Wiley & Sons, West Sussex, England, 2002.

L. Diao, K. Hu, Y. Lu, and Ch. Shi. A method to boost support vector machines. In M-S. Chen, P. S. Yu, and B. Liu, editors, *Proceedings of the Sixth Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 463–468, Taipei, Taiwan, 2002. Springer-Verlag.

T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40:139–157, 2000a.

T. G. Dietterich. Ensemble methods in machine learning. In J. Kittler and F. Roli, editors, *Proceedings of the First International Workshop on Multiple Classifier Systems*, pages 1–15. Springer-Verlag, 2000b.

T. G. Dietterich. Ensemble methods in machine learning. *Lecture Notes in Computer Science*, 1857: 1–15, 2000c.

T. G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923, 1998.

C. Domingo and O. Watanabe. MadaBoost: A modification of AdaBoost. In *Proceedings of the 13th Annual Conference on Computational Learning Theory*, pages 180–189. Morgan Kaufmann, San Francisco, 2000.

S. Dzeroski and B. Zenko. Is combining classifiers with stacking better than selecting the best one? *Machine Learning*, 54:255–273, 2004.

G. Eibl and K-P. Pfeiffer. Multiclass boosting for weak classifiers. *Journal of Machine Learning Research*, 6:189–210, 2005.

A. Fern and R. Givan. Online ensemble learning: An empirical study. *Machine Learning*, 53: 71–109, 2003.

Y. Freund and R. Schapire. Experiments with a new boosting algorithm. In *Proc. of the Thirteenth International Conference on Machine Learning*, pages 148–156, Bari, Italy, 1996.

J. Friedman, T. Hastie, and R. Tibshirani. Additice logistic regression: A statistical view of boosting. *Annals of Statistics*, 28(2):337–407, 2000.

N. García-Pedrajas, C. Hervás-Martínez, and D. Ortiz-Boyer. Cooperative coevolution of artificial neural network ensembles for pattern classification. *IEEE Transactions on Evolutionary Computation*, 9(3):271–302, June 2005.

R. L. Gorsuch. *Factor Analysis*. Erlbaum, Hillsdale, NJ, USA, 1983.

A. J. Grove and D. Schuurmans. Boosting in the limit: Maximizing the margin of learned ensembles. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 1998.

L. Hall, K. Bowyer, R. Banfield, D. Bhadoria, W. Kegelmeyer, and S. Eschrich. Comparing pure parallel ensemble creation techniques against bagging. In *Third IEEE International Conference on Data Mining*, pages 533–536, Melbourne, FL, USA, 2003.

S. Haykin. *Neural Networks – A Comprehensive Foundation*. Prentice – Hall, Upper Saddle River, NJ, 2nd edition, 1999.

S. Hettich, C.L. Blake, and C.J. Merz. UCI repository of machine learning databases, 1998. http://www.ics.uci.edu/∼mlearn/MLRepository.html.

T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

I. T. Jolliffe. *Principal Components Analysis*. Springer – Verlag, New York, NY, 1986.

H-Ch. Kim, S. Pang, H-M. Je, D. Kim, and S. Y. Bang. Pattern classification using support vector machine ensembles. In *Proceedings of the 16th International Conference on Pattern Recognition (ICPR´02)*, volume 2, pages 160–163, 2002.

E. Kleinberg. On the algorithmic implementation of stochastic discrimination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(5):473–490, 2000.

R. Kohavi. *Wrappers for Performance Enhancement and Oblivious Decision Graphs*. PhD thesis, Department of Computer Science, Stanford University, Stanford, USA, 1995.

R. Kohavi and C. Kunz. Option decision trees with majority voting. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 161–169, San Francisco, CA, USA, 1997. Morgan Kaufman.

T. Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, Berlin, third edition, 2001.

J. F. Kolen and J. B. Pollack. Back propagation is sensitive to initial conditions. In Richard P. Lippmann, John E. Moody, and David S. Touretzky, editors, *Advances in Neural Information Processing Systems*, volume 3, pages 860–867. Morgan Kaufmann Publishers, Inc., 1991.

L. Kuncheva and C. J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181–207, May 2003.

L. I. Kuncheva. Combining classifiers: Soft computing solutions. In S. K. Pal and A. Pal, editors, *Pattern Recognition: From Classical to Modern Approaches*, pages 427–451. World Scientific, 2001.

L. I. Kuncheva. Error bounds for aggressive and conservative adaboost. In *Proceedings of MCS*, number 2709 in Lecture Notes in Computer Science, pages 25–34, Guilford, UK, 2003.

Y. LeCun, L. Bottou, G. B. Orr, and K-R. Müller. Efficient backprop. In G. B. Orr and K-R. Müller, editors, *Neural Networks: Tricks of the Trade*, pages 9–50. Springer-Verlag, 1998.

B. Lerner, H. Guterman, M. Aladjem, and I. Dinstein. A comparative study of neural networks based feature extraction paradigms. *Pattern Recognition Letters*, 20(1):7–14, 1999.

Y. Liu, X. Yao, and T. Higuchi. Evolutionary ensembles with negative correlation learning. *IEEE Transactions on Evolutionary Computation*, 4(4):380–387, November 2000.

D. D. Margineantu and T. G. Dietterich. Pruning adaptive boosting. In Douglas H. Fisher, editor, *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 211–218, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

L. Mason, P. L. Bartlett, and J. Baxter. Improved generalization through explicit optimization of margins. *Machine Learning*, 38:243–255, 2000.

C. J. Merz. Using correspondence analysis to combine classifiers. *Machine Learning*, 36(1):33–58, July 1999.

R. Munro, D. Ler, and J. Patrick. Meta-learning orthographic and contextual models for language independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning*, pages 192 – 195, 2003.

D. W. Opitz. Feature selection for ensembles. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pages 379 – 384, Orlando, FL, USA, 1999. American Association for Artificial Intelligence.

D. Ortiz-Boyer, C. Hervás-Martínez, and N. García-Pedrajas. Cixl2: A crossover operator for evolutionary algorithms based on population features. *Journal of Artificial Intelligence Research*, 24:33–80, July 2005.

J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, 1993.

J. J. Rodríguez, L. I. Kuncheva, and C. J. Alonso. Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1619–1630, Oct 2006.

S. Rosset, J. Zhu, and T. Hastie. Boosting as a regularized path to a maximum margin classifier. *Journal of Machine Learning Research*, 5:941–073, 2004.

D. Rumelhart, G. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. Rumelhart and J. McClelland, editors, *Parallel Distributed Processing*, pages 318–362. MIT Press, Cambridge, MA, 1986.

R. E. Schapire and Y. Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39:135–168, 2000.

R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37:297–336, 1999.

R. E. Schapire, Y. Freund, P. L. Bartlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics*, 26(5):1651–1686, 1998.

M. Sebban, R. Nock, and S. Lallich. Stopping criterion for boosting-based data reduction techniques: from binary to multiclass problems. *Journal of Machine Learning Research*, 3:863–885, 2002.

M. Skurichina and R. P. W. Duin. Bagging and the random subspace method for redundant feature spaces. In J. Kittler and R. Poli, editors, *Proceedings of the Second International Workshop on Multiple Classifier Systems MCS 2001*, pages 1–10, Cambridge, UK, 2001.

A. Tsymbal, P. Cunningham, M. Pechinizkiy, and P. Puuronen. Search strategies for ensemble feature selection in medical diagnosis. In M. Krol, S. Mitra, and D. J. Lee, editors, *Proceedings of the Sixteenth IEEE Symposium on Computer-Bases Medical Systems CBMS'2003*, pages 124–129, The Mount Sinai School of Medicine, New York, USA, 2003. IEEE CS Press.

K. Tumer and J. Ghosh. Error correlation and error reduction in ensemble classifier. *Connection Science*, 8(3–4):385–404, 1996.

A. Utsugi. Ensemble of independent factor analyzers with application to natural image analysis. *Neural Processing Letters*, 14(1):49–60, August 2001.

G. I. Webb. Multiboosting: A technique for combining boosting and wagging. *Machine Learning*, 40(2):159–196, August 2000.

M-H. Yand, N. Ahuja, and D. Kriegman. Face detection using mixtures of linear subspaces. In *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, pages 70–77. IEEE Computer Society Washington, DC, USA, 2000.

G. Zenobi and P. Cunningham. Using diversity in preparing ensembles of classifiers based on different feature subsets to minimize generalization error. In L. de Raedt and P. Flach, editors, *12th European Conference on Machine Learning (ECML 2001)*, LNAI 2167, pages 576–587. Springer–Verlag, 2001.

T. Zhang and B. Yu. Boosting with early stooping: Convergence and consistency. *The Annals of Statistics*, 33(4):1538–1579, 2005.

Z-H. Zhou, J. Wu, and W. Tang. Ensembling neural networks: Many could be better than all. *Artificial Intelligence*, 137(1–2):239–253, May 2002.

Journal of Machine Learning Research 8 (2007) 35-63 Submitted 4/06; Revised 9/06; Published 1/07

Multi-Task Learning for Classification with Dirichlet Process Priors

Ya Xue YX10@EE.DUKE.EDU
Xuejun Liao XJLIAO@EE.DUKE.EDU
Lawrence Carin LCARIN@EE.DUKE.EDU
Department of Electrical and Computer Engineering
Duke University
Durham, NC 27708, USA

Balaji Krishnapuram BALAJI.KRISHNAPURAM@SIEMENS.COM
Siemens Medical Solutions USA, Inc.
Malvern, PA 19355, USA

Editor: Peter Bartlett

Abstract

Consider the problem of learning logistic-regression models for multiple classification tasks, where the training data set for each task is not drawn from the same statistical distribution. In such a multi-task learning (MTL) scenario, it is necessary to identify groups of similar tasks that should be learned jointly. Relying on a Dirichlet process (DP) based statistical model to learn the extent of similarity between classification tasks, we develop computationally efficient algorithms for two different forms of the MTL problem. First, we consider a *symmetric* multi-task learning (SMTL) situation in which classifiers for multiple tasks are learned jointly using a variational Bayesian (VB) algorithm. Second, we consider an *asymmetric* multi-task learning (AMTL) formulation in which the posterior density function from the SMTL model parameters (from previous tasks) is used as a prior for a new task: this approach has the significant advantage of not requiring storage and use of all previous data from prior tasks. The AMTL formulation is solved with a simple Markov Chain Monte Carlo (MCMC) construction. Experimental results on two real life MTL problems indicate that the proposed algorithms: (a) automatically identify subgroups of related tasks whose training data appear to be drawn from similar distributions; and (b) are more accurate than simpler approaches such as single-task learning, pooling of data across all tasks, and simplified approximations to DP.

Keywords: classification, hierarchical Bayesian models, Dirichlet process

1. Introduction

A real world classification task can often be viewed as consisting of multiple correlated subtasks. In remote sensing, for example, one may have multiple sets of data, each collected at a particular geographical location; rather than designing individual classifiers for each of these sensing tasks, it is desirable to share data across tasks to enhance overall sensing performance. This represents a typical example of a general learning scenario called multi-task learning (MTL) (Caruana, 1997), or learn to learn (Thrun and Pratt, 1998). In contrast to MTL, single-task learning (STL) refers to the approach of learning one classification task at a time, only using the corresponding data set; often STL assumes that the training samples are drawn independently from an identical distribution. MTL is distinct from standard STL in two principal respects: (i) the tasks are not identical, thus simply pooling them and treating them as a single task is not appropriate; and (ii) some of the classification tasks may be highly correlated (dependent on each other), but the strategy of isolating each task and learning the corresponding classifier independently does not exploit the potential information one may acquire from other classification tasks.

The fact that some of the classification tasks are correlated (dependent) implies that what is learned from one task is transferable to another. By learning the classifiers in parallel under a unified representation, the transferability of expertise between tasks is exploited to the benefit of all. This expertise transfer is particularly important when we are provided with only a limited amount of training data for learning each classifier. By exploiting data from related tasks, the training data for each task is strengthened and the generalization of the resulting classifier-estimation algorithm is improved.

1.1 Previous Work on MTL

Multi-task learning has been the focus of much interest in the machine learning community over the last decade. Typical approaches to information transfer among tasks include: sharing hidden nodes in neural networks (Baxter, 1995, 2000; Caruana, 1997); placing a common prior in hierarchical Bayesian models (Yu et al., 2003, 2004, 2005; Zhang et al., 2006); sharing parameters of Gaussian processes (Lawrence and Platt, 2004); learning the optimal distance metric for K-Nearest Neighbors (Thrun and O'Sullivan, 1996); sharing a common structure on the predictor space (Ando and Zhang, 2005); and structured regularization in kernel methods (Evgeniou et al., 2005), among others.

In statistics, the problem of combining information from similar but independent experiments has been studied under the category of meta-analysis (Glass, 1976) for a variety of applications in medicine, psychology and education. Researchers collect data from experiments performed at different sites or times—including information from related experiments published in literature—to obtain an overall evaluation on the significance of an experimental effect. Therefore, meta-analysis is also referred to as quantitative synthesis, or overview. The objective of multi-task learning is different from that of meta analysis. Instead of giving an overall evaluation, our objective is to learn multiple tasks jointly, either to improve the learning performance (i.e., classification accuracy) of each individual task, or to boost the performance of a new task by transferring domain knowledge learned from previously observed tasks. Despite the difference in objectives, many of the techniques employed in the statistical literature on meta-analysis can be applied to multi-task learning as well.

1.1.1 DIRICHLET PROCESSES FOR NONPARAMETRIC HIERARCHICAL BAYESIAN MODELING

Hierarchical Bayesian modeling is one of the most important methods for meta analysis (Burr and Doss., 2005; Dominici et al., 1997; Hoff, 2003; Müller et al., 2004; Mallick and Walker, 1997). Hierarchical Bayesian models provide the flexibility to model both the individuality of tasks (experiments), and the correlations between tasks. Statisticians refer to this approach as "borrowing strength" across tasks. Usually the bottom layer of the hierarchy is individual models with task-specific parameters. On the layer above, tasks are connected together via a common prior placed on those parameters. The hierarchical model can achieve efficient information-sharing between tasks for the following reason. Learning of the common prior is also a part of the training process, and data from all tasks contribute to learning the common prior, thus making it possible to transfer information between tasks (via sufficient statistics). Given the prior, individual models are learnt independently. As a result, the estimation of a classifier (task) is affected by both its own training data and by data from the other tasks related through the common prior.

Often, the common prior in a hierarchical Bayesian model is specified in a parametric form with unknown hyper-parameters, for example, a Gaussian distribution with unknown mean and variance. Information is transferred between tasks by learning those hyper-parameters using data from all tasks. However, it is preferable to also learn the functional form of the common prior from the data, instead of being pre-defined. In this paper, we provide such a nonparametric hierarchical Bayesian model for jointly learning multiple logistic regression classifiers. Such nonparametric approaches are desirable because it is often difficult to know what the true distribution should be like, and an inappropriate prior could be misleading. Further, the model parameters of individual tasks may have high complexity, and therefore no appropriate parametric form can be found easily.

In the proposed nonparametric hierarchical Bayesian model, the common prior is drawn from the Dirichlet process (DP). The advantage of applying the DP prior to hierarchical models has been addressed in the statistics literature, see for example Mukhopadhyay and Gelfand (1997), Mallick and Walker (1997) and Müller et al. (2004). Research on the Dirichlet process model goes back to Ferguson (1973), who proved that there is positive (non-zero) probability that some sample function of the DP will be as close as desired to any probability function defined on the same support set. Therefore, the DP is rich enough to model the parameters of individual tasks with arbitrarily high complexity, and flexible enough to fit them well without any assumption about the functional form of the prior distribution.

1.1.2 IDENTIFYING THE EXTENT OF SIMILARITIES BETWEEN TASKS IN MTL

A common assumption in the previous literature on MTL work is that all tasks are (equally) related to each other, but recently there have been a few investigations concerning the extent of relatedness between tasks. An ideal MTL algorithm should be able to automatically identify similarities between tasks and only allow similar tasks to share data or information. Thrun and O'Sullivan (1996) first presented a task-clustering algorithm with K-Nearest Neighbors. Bakker and Heskes (2003) model the common prior in the hierarchical model as a mixture distribution, but two issues exist in that work: (i) Extra "high-level" task characteristics, other than the features used for learning the model parameters of individual tasks, are needed to decide the relative weights of mixture components; and (ii) the number of mixtures is a pre-defined parameter. Both these issues are avoided in the models presented here. Based only on the features and class labels, the proposed statistical models automatically identify the similarities between the various tasks and adjust the complexity of the model, that is, the number of task clusters, relying on the implicit nonparametric clustering mechanism of the DP (see Sec. 2).

Before we proceed with the technical details, we clarify two issues about task-similarity. First, we define two classification tasks as similar when the two classification boundaries are close, that is, when the weight vectors of two classifiers are similar. Note that this is different from some previous work such as Caruana (1997) where two tasks are defined to be similar if they use the same features to make their decision.

Secondly, the property that the distributions drawn from a Dirichlet process are discrete with probability one introduces questions, because it implies that we cluster *identical* tasks instead of *similar* tasks. This may appear restricting, but for the following reasons this is not the case. We are interested in the posterior distribution of the model parameters when we learn a model with Bayesian methods. The posterior is decided by both the prior of the parameters and the data likelihood given the parameters. If the DP prior is employed, the prior promotes clustering while the likelihood encourages the fitting of the parameters of individual classifiers to their own data. Therefore, the model parameters learned for any task are the result of the tradeoff between sharing with other tasks and retaining the individuality of the current task. This gives similar tasks the chance to share the same model parameters.

As discussed above, the direct use of DP yields identical parameter estimates to similar tasks. One may more generally wish to give similar (but not identical) parameters to similar tasks. This may be accomplished by adding an additional layer of randomness, that is, modeling the prior of the model parameters as a DP mixture (DPM) (Antoniak, 1974), which is a usual solution to the discrete character of DP. We have compared the models with the DPM prior and the DP prior and found that the former underperforms the latter in all our empirical experiments. We hypothesize that this result is attributable to the increase in model complexity, since the DPM prior introduces one more layer in the hierarchical model than the DP prior. Therefore, in this paper we only present the approach of placing a DP prior on the the model parameters of individual tasks.

1.2 Novel Contributions of this Paper

We develop novel solutions to the following problems: (i) joint learning of multiple classification tasks, which may differ in data statistics due to temporal, geographical or other variations; (ii) efficient transfer of the information learned from (i) to a new observed task for the purpose of improving the new task's learning performance. For notational simplification, problem (i) is referred as SMTL (symmetric multi-task learning) and problem (ii) as AMTL (asymmetric multi-task learning). Our discussion is focused on classification tasks, although the proposed methods can be extended directly to regression problems.

The setting of the SMTL is similar to that used in meta analysis. Most meta-analysis work considers only simple linear models for regression problems. Mukhopadhyay and Gelfand (1997) and Ishwaran (2000) discuss the application of DP priors in the general framework of generalized linear models (GLMs). They use the mixed random effects model to capture heterogeneity in the population studied in an experiment. The fixed effects are modeled with a parametric prior while the random effects are mixed over DP. Our work is closely related to their research, in that the logistic regression model, which we use for classification, is a special case of GLMs. Yet, we modify the model to suit the multi-task setting. First, we group the population by task and add an additional constraint that each group share the same model. Second, we place a DP prior on the whole vector of linear coefficients, including both fixed effects and random effects. The linear coefficients of covariates in the logistic regression model correspond to the classifiers in linear classification problems. For our problem, it is too limiting to consider only the variability in the random effect, that is, the intercept of the classification boundary; the variability in the orientation of the classification boundary should be considered as well.

The inference techniques employed here constitute a major difference between our SMTL part and the work of Mukhopadhyay and Gelfand (1997) and Ishwaran (2000). Mukhopadhyay and Gelfand (1997) use Gibbs sampling based on the Polya Urn representation of DP, while Ishwaran (2000) develops a Gibbs sampler based on the truncated stick-breaking representation of DP. In the work reported here, the proposed SMTL models are implemented with a deterministic inference method, specifically variational Bayesian (VB) inference, avoiding the overfitting associated with maximum-likelihood (ML) approximations while preserving computational efficiency.

The AMTL problem addressed here is a natural extension of SMTL. In AMTL the computational burden is relatively small and thus a means of Monte Carlo integration with acceptance-rejection is suggested to make predictions for the new task. We note that Yu et al. (2004) propose a hierarchical Bayesian framework for information filtering, which similarly applies a DP prior on individual user profiles. However, due to limitations of the approximate DP prior employed in Yu et al. (2004), their approach differs from ours in two respects: (i) it cannot be used to improve the classification performance of multiple tasks by learning them in parallel, that is, it is not a solution to the SMTL problem; and (ii) in the AMTL case, their approach conducts an effective information transfer to the new observed task only when the size of the training set is large in previous tasks. These points are clarified further in Section 4.

1.3 Organization of the Paper

The remainder of this paper is organized as follows. Section 2 provides an introduction to the Dirichlet process and its application to data clustering. The SMTL problem is addressed in Section 3, which describes the proposed Bayesian hierarchical framework and presents a variational inference algorithm. Section 4 develops an efficient method of information transfer for the AMTL case and compares it with the method presented by Yu et al. (2004). Experimental results are reported in Section 5, demonstrating the application of the proposed models to a landmine detection and an art image retrieval problem. Section 6 concludes the work and outlines future directions.

2. Dirichlet Process

Assume the model parameters of individual tasks, denoted by w, are drawn from a common prior distribution G. The distribution G itself is sampled from a Dirichlet process $DP(\alpha, G_0)$, where α is a positive scaling (innovation) parameter and G_0 is a base distribution. The mathematical representation of the DP model is

$$\begin{aligned} w_m|G &\sim G, \\ G &\sim DP(\alpha, G_0). \end{aligned}$$

where $m = 1, \dots, M$ for M tasks.

Integrating out G, the conditional distribution of w_m, given observations of the other $M-1$ w values $w_{-m} = \{w_1, \cdots, w_{m-1}, w_{m+1}, \cdots, w_M\}$, is

$$p(w_m|w_{-m}, \alpha, G_0) = \frac{\alpha}{M-1+\alpha} G_0 + \frac{1}{M-1+\alpha} \sum_{j=1, j \neq m}^{M} \delta_{w_j}, \tag{1}$$

where δ_{w_j} is the distribution concentrated at the single point w_j.

Let w_k^*, $k = 1, \dots, K$, denote the K distinct values among $w_1, \dots, w_M$ and $n_{-m,k}$ denote the number of w's equal to w_k^*, excluding w_m. Equation (1) can be rewritten as

$$p(w_m|w_{-m}, \alpha, G_0) = \frac{\alpha}{M-1+\alpha} G_0 + \frac{1}{M-1+\alpha} \sum_{k=1}^{K} n_{-m,k} \delta_{w_k^*}. \tag{2}$$

Relevant observations concerning (2) are: (i) the Dirichlet process model has an implicit mechanism of clustering samples into groups, since a new sample prefers to join a group with a large population; (ii) the model is flexible, creating new clusters or merging existing clusters to fit the observed data better; (iii) parameter α controls the probability of creating a new cluster, with larger α yielding more clusters; (iv) in the limit as $\alpha \to \infty$ there is a cluster for each w, and since w are drawn from G_0, $\lim_{\alpha\to\infty} G = G_0$.

The Dirichlet Process has the following important properties:

1. $E[G] = G_0$; the base distribution represents our prior knowledge or expectation concerning G.
2. $p(G|w_1,\ldots,w_M,\alpha,G_0) = DP(\alpha+M, \frac{\alpha}{M+\alpha}G_0 + \frac{1}{M+\alpha}\sum_{j=1}^{M}\delta_{w_j})$; the posterior of G is still a Dirichlet process with the updated base distribution $\frac{\alpha}{M+\alpha}G_0 + \frac{1}{M+\alpha}\sum_{j=1}^{M}\delta_{w_j}$, and a confidence in this base distribution that is enhanced relative to the confidence in the original base distribution G_0, reflected in the increased parameter $\alpha \to \alpha+M$.

The lack of an explicit form of G is addressed with the stick-breaking view of the Dirichlet process. Sethuraman (1994) introduces a constructive definition of the Dirichlet process, based upon which Ishwaran and James (2001) characterize the DP priors with a stick-breaking representation

$$G = \sum_{k=1}^{\infty} \pi_k \delta_{w_k^*}, \tag{3}$$

where

$$\pi_k = v_k \prod_{i=1}^{k-1}(1-v_i).$$

For each k, v_k is drawn from a Beta distribution $Be(1,\alpha)$;[1] simultaneously another random variable w_k^* is drawn independently from the base distribution G_0; w_k^* and π_k represents the location and weight of the kth stick.

If v_K is set to one instead of being drawn from the beta distribution, it yields a truncated approximation to the Dirichlet process

$$G = \sum_{k=1}^{K} \pi_k \delta_{w_k^*}.$$

Ishwaran and James (2001) establish two theorems for selecting an appropriate truncation level, leading to a model virtually indistinguishable from the infinite DP model; the truncated DP model is computationally more efficient in practice.

3. Learning Multiple Classification Tasks Jointly (SMTL)

In this section we propose a Bayesian multi-task learning model for jointly estimating classifiers for several data sets. The model automatically identifies relatedness by task clustering with nonparametric methods. A variational Bayesian (VB) approximation is used to learn the posterior distributions of the model parameters.

1. Notation for distributions follows Robert and Casella (2004); this same notation is used in the rest of the paper.

3.1 Mathematical Model

Consider M tasks, indexed as $1,\cdots,M$. Let the data set of task m be $\mathcal{D}_m = \{(x_{m,n}, y_{m,n}) : n = 1,\cdots,N_m\}$, where $x_{m,n} \in \mathbb{R}^d$, $y_{m,n} \in \{0,1\}$, and $(x_{m,n}, y_{m,n})$ are drawn i.i.d. from the underlying distribution of task m. For task m the conditional distribution of $y_{m,n}$ given $x_{m,n}$ is modeled via logistic regression as,

$$p(y_{m,n}|w_m, x_{m,n}) = \sigma(w_m^T x_{m,n})^{y_{m,n}}[1-\sigma(w_m^T x_{m,n})]^{1-y_{m,n}} \tag{4}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ and w_m parameterizes the classifier for task m. The goal is to learn $\{w_m\}_{m=1}^M$ jointly, sharing information between tasks as appropriate, so that the resulting classifiers can accurately predict class labels for new test samples for tasks $m = 1,\cdots,M$.

To complete the hierarchical model, we place a Dirichlet process prior on the parameters w_m, which based on the discussion in Section 2 implies clustering of the tasks. The base distribution G_0 is specified as a d-dimensional multivariate normal distribution $N_d(\mu,\Sigma)$. We define an indicator variable $c_m = [c_{m,1},\ldots,c_{m,\infty}]^T$, which is an all-zero vector except that the kth entry is equal to one if task m belongs to cluster k, using the infinite set of mixture components (clusters) reflected in (3). The data can be seen as drawn from the following generative model, obtained by employing the stick-breaking view of DP:

SMTL Model. Given the parameters α, μ and Σ,

1. Draw v_k from the Beta distribution $Be(1,\alpha)$ and independently draw w_k^* from the base distribution $N_d(\mu,\Sigma)$, $k = 1,\ldots,\infty$.
2. $\pi_k = v_k \prod_{i=1}^{k-1}(1-v_i), k = 1,\ldots,\infty$.
3. Draw the indicators $(c_{m,1},\ldots,c_{m,\infty})$ from a multinomial distribution $M_\infty(1;\pi_1,\ldots,\pi_\infty)$, $m = 1,\ldots,M$
4. $w_m = \prod_{k=1}^{\infty}(w_k^*)^{c_{m,k}}$, or in an equivalent form $w_m = \sum_{k=1}^{\infty} c_{m,k} w_k^*$, $m = 1,\ldots,M$.
5. Draw $y_{m,n}$ from a Binomial distribution $B(1,\sigma(w_m^T x_{m,n}))$, $m = 1,\ldots,M$, $n = 1,\cdots,N_m$.

We refer to this as symmetric multi-task learning (SMTL) because all tasks are treated symmetrically; asymmetric multi-task learning (AMTL) is addressed in Section 4.

3.1.1 HYPER-PRIORS

In the SMTL model, α, μ and Σ are given parameters. The scaling parameter α often has a strong impact on the number of clusters, as analyzed in Section 2. To make the algorithm more robust, it is suggested by West et al. (1994) that α be integrated over a diffuse hyper-prior. This leads to a modified SMTL model:

SMTL-1 Model. Given the parameters μ, Σ and hyper-parameters τ_{10}, τ_{20},

- Draw α from a Gamma distribution $Ga(\tau_{10},\tau_{20})$.
- Follow Step 1-5 in the SMTL model.

Similarly, we are also interested in the effects of integrating μ and Σ, the parameters of the base distribution, over a diffuse prior. For notational simplification, we use Λ, the precision matrix of the base distribution, instead of the covariance matrix Σ, where $\Lambda = \Sigma^{-1}$. We assume Λ is a diagonal matrix with diagonal elements $\lambda_1, \cdots, \lambda_d$. The conjugate prior for the mean and precision of a normal distribution is a Normal-Gamma distribution. Hence, the SMTL-1 model can be further modified as

SMTL-2 Model. Given the hyper-parameters $\tau_{10}, \tau_{20}, \gamma_{10}, \gamma_{20}$ and β_0,

- Draw λ_j from a Gamma distribution $Ga(\gamma_{10}, \gamma_{20})$, $j = 1, ..., d$.
- Draw μ from a Normal distribution $N_d(\mathbf{0}, (\beta_0\Lambda)^{-1})$; draw α from a Gamma distribution $Ga(\tau_{10}, \tau_{20})$.
- Follow Step 1-5 in the SMTL model.

The performance of the SMTL-1 and SMTL-2 models is analyzed in Section 5.1.1.

3.1.2 Graphical Representation

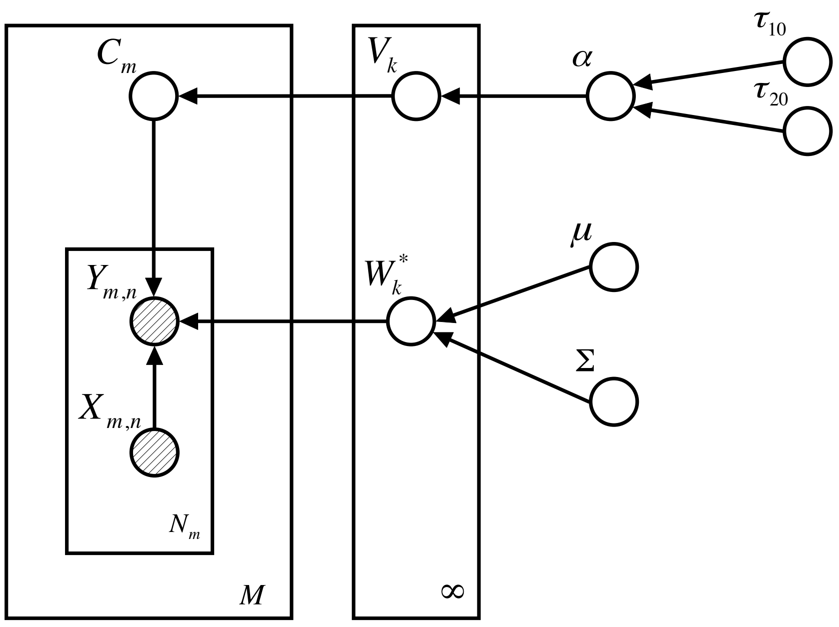
Figure 1 shows a graphical representation of the SMTL models with the Dirichlet process prior; the two SMTL models differ in the ancestor nodes of w_k^*.

In the graph, each node denotes a random variable in the model. The nodes for $y_{m,n}$ and $x_{m,n}$ are shaded because they are observed data. An arrow indicates dependence between variables, that is, the conditional distribution of a variable given its parents. The number, for example, M, at the lower right corner of a box indicates the nodes in that box have M *iid* copies.

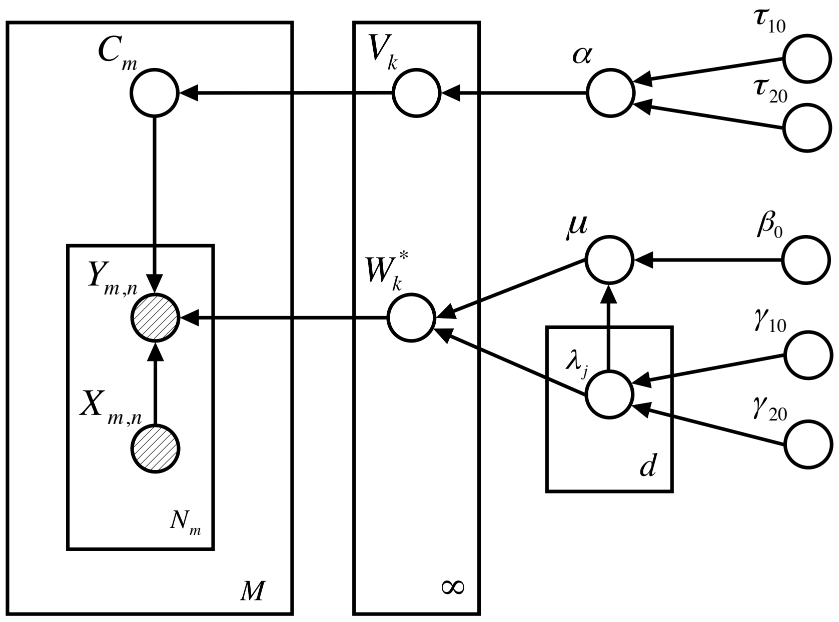
The condition distributions between the variables are specified as follows

- $y_{m,n}$
  $p(y_{m,n}|c_m, \{w_k^*\}_{k=1}^{\infty}, x_{m,n}) = \prod_{k=1}^{\infty} \{\sigma(w_k^{*T} x_{m,n})^{y_{m,n}} [1-\sigma(w_k^{*T} x_{m,n})]^{1-y_{m,n}}\}^{c_{m,k}}$,
  $m = 1, \ldots, M, n = 1, \cdots, N_m$.
- c_m
  $p(c_m|\{v_k\}_{k=1}^{\infty}) = v_1^{c_{m,1}} \prod_{k=2}^{\infty} [v_k \prod_{i=1}^{k-1} (1-v_i)]^{c_{m,k}}, m = 1, \ldots, M.$
- v_k
  $p(v_k|\alpha) = \alpha(1-v_k)^{\alpha-1}, k = 1, \ldots, \infty,$
- α
  $p(\alpha|\tau_{10}, \tau_{20}) = \frac{\tau_{20}^{\tau_{10}}}{\Gamma(\tau_{10})} \alpha^{\tau_{10}-1} \exp(-\tau_{20}\alpha)$, where $\Gamma(\cdot)$ is the Gamma function.
- w_k^*
  - SMTL-1 Model:
    $p(w_k^*|\mu, \Sigma) = (2\pi)^{-\frac{d}{2}} |\Sigma|^{-\frac{1}{2}} \exp(-\frac{1}{2}(w_k^* - \mu)^T \Sigma^{-1} (w_k^* - \mu)), k = 1, \ldots, \infty.$
  - SMTL-2 Model:
    $p(w_k^*|\mu, \{\lambda_j\}_{j=1}^d) = (2\pi)^{-\frac{d}{2}} |\Lambda|^{\frac{1}{2}} \exp(-\frac{1}{2}(w_k^* - \mu)^T \Lambda (w_k^* - \mu)), k = 1, \ldots, \infty,$
    where Λ is a diagonal matrix with diagonal elements $\lambda_1, \cdots, \lambda_d$.
- $\mu, \lambda_1, \cdots, \lambda_d$ (if the SMTL-2 Model is used)

$$\begin{aligned} & p(\mu, \{\lambda_j\}_{j=1}^d | \gamma_{10}, \gamma_{20}, \beta_0) \\ = \; & (2\pi)^{-\frac{d}{2}} |\beta_0\Lambda|^{\frac{1}{2}} \exp(-\tfrac{\beta_0}{2}\mu^T \Lambda \mu) \cdot \prod_{j=1}^{d} [\tfrac{\gamma_{20}^{\gamma_{10}}}{\Gamma(\gamma_{10})} \lambda_j^{\gamma_{10}-1} \exp(-\gamma_{20}\lambda_j)]. \end{aligned}$$



(a) Graphical representation of the SMTL-1 model.



(b) Graphical representation of the SMTL-2 model.

Figure 1: Graphical representation of the SMTL models with the Dirichlet process prior.

For simplicity, let $\mathbf{Z}$ denote the collection of latent variables and Φ denote the collection of given parameters and hyper-parameters. For the SMTL-1 model, $\mathbf{Z} = \{\{c_m\}_{m=1}^M, \{v_k\}_{k=1}^\infty, \alpha, \{w_k^*\}_{k=1}^\infty\}$

and $\Phi = \{\tau_{10}, \tau_{20}, \mu, \Sigma\}$; for the SMTL-2 model, $\mathbf{Z} = \{\{c_m\}_{m=1}^M, \{v_k\}_{k=1}^\infty, \alpha, \{w_k^*\}_{k=1}^\infty, \mu, \{\lambda_j\}_{j=1}^d\}$ and $\Phi = \{\tau_{10}, \tau_{20}, \gamma_{10}, \gamma_{20}, \beta_0\}$.

3.2 Variational Bayesian Inference

In the Bayesian approach, we are interested in $p(\mathbf{Z}|\{\mathcal{D}_m\}_{m=1}^M, \Phi)$, the posterior distribution of the latent variables given the observed data and hyper-parameters,

$$p(\mathbf{Z}|\{\mathcal{D}_m\}_{m=1}^M, \Phi) = \frac{p(\{\mathcal{D}_m\}_{m=1}^M|\mathbf{Z}, \Phi)p(\mathbf{Z}|\Phi)}{p(\{\mathcal{D}_m\}_{m=1}^M|\Phi)},$$

where $p(\{\mathcal{D}_m\}_{m=1}^M|\Phi) = \int p(\{\mathcal{D}_m\}_{m=1}^M|\mathbf{Z}, \Phi)p(\mathbf{Z}|\Phi)dZ$ is the marginal distribution, and evaluation of this integration is the principal computational challenge; the integral does not have an analytic form in most cases. The Markov Chain Monte Carlo (MCMC) method is a powerful and popular simulation tool for Bayesian inference. To date most research on applications of the Dirichlet process has been implemented with Gibbs sampling, an MCMC method (Escobar and West, 1995; Ishwaran and James, 2001). However, slow speed and difficult-to-evaluate convergence of the DP Gibbs sampler impede its application in many practical situations.

In this work, we employ a computationally efficient approach, mean-field variational Bayesian (VB) inference (Ghahramani and Beal, 2001). The VB method approximates the true posterior $p(\mathbf{Z}|\{\mathcal{D}_m\}_{m=1}^M, \Phi)$ by a variational distribution $q(\mathbf{Z})$. It converts computation of posteriors into an optimization problem of minimizing the Kullback-Leibler (KL) distance between $q(\mathbf{Z})$ and $p(\mathbf{Z}|\{\mathcal{D}_m\}_{m=1}^M, \Phi)$, which is equivalent to maximizing a lower bound of $\log p(\{\mathcal{D}_m\}_{m=1}^M|\Phi)$, the log likelihood. To make the optimization problem tractable, it is assumed that the variational distribution $q(\mathbf{Z})$ is sufficiently simple - fully factorized with each factorized component in the exponential family. Under this assumption, an analytic solution of the optimal $q(\mathbf{Z})$ can be obtained by taking functional derivatives; refer to Ghahramani and Beal (2001) and Jordan et al. (1999) for more details about VB.

3.2.1 Local Convex Bound

One difficulty of applying VB inference to the SMTL models is that the sigmoid function in (4) does not lie within the conjugate-exponential family. We use a variational method based on bounding log convex functions (Jaakkola and Jordan, 1997).

Consider the logistic regression model $p(y|w,x) = \sigma(w^Tx)^y[1-\sigma(w^Tx)]^{1-y}$. The prior distribution of w is assumed to be normal with mean $\tilde{\mu}_0$ and variance $\tilde{\Sigma}_0$. We want to estimate the posterior distribution of w given the data (x,y). This does not have an analytic solution due to the non-exponential property of the logistic regression function. Jaakkola and Jordan (1997) present a method that uses an accurate variational transformation of $p(y|w,x)$ as follows

$$p(y|w,x) \geq \sigma(\xi)\exp(\frac{(2y-1)w^Tx - \xi}{2} + \rho(\xi)(x^Tww^Tx - \xi^2)),$$

where $\rho(\xi) = \frac{\frac{1}{2}-\sigma(\xi)}{2\xi}$ and ξ is a variational parameter. The equality holds when $\xi = \pm w^Tx$.

The posterior $p(w|x,y,\tilde{\mu}_0,\tilde{\Sigma}_0)$ remains the normal form with this variational approximation. Given the variational parameter ξ, the mean $\tilde{\mu}$ and variance $\tilde{\Sigma}$ of the normal distribution can be computed as

$$\tilde{\Sigma} = (\tilde{\Sigma}_0^{-1} + 2|\rho(\xi)|xx^T)^{-1}, \quad \tilde{\mu} = \tilde{\Sigma}[\tilde{\Sigma}_0^{-1}\tilde{\mu}_0 + (y - \frac{1}{2})x]. \tag{5}$$

Note that we use the top script $\tilde{}$ to avoid confusion with the usage of μ and Σ as the mean and variance of the DP base distribution G_0 in other sections of this paper.

Since the optimal value of ξ depends on w, an EM algorithm is devised in Jaakkola and Jordan (1997) to optimize ξ. The E step updates $\tilde{\mu}$ and $\tilde{\Sigma}$ following (5), given the estimate of ξ in the last iteration; the M step computes the optimal value of ξ as

$$\xi^2 = x^T(\tilde{\Sigma} + \tilde{\mu}\tilde{\mu}^T)x.$$

This EM algorithm is assured to converge and has been verified to be fast and stable in Jaakkola and Jordan (1997).

The variational method can give us a lower bound of the predictive likelihood as

$$p(y|x,\tilde{\mu}_0,\tilde{\Sigma}_0) \geq \exp(\log\sigma(\xi) - \frac{\xi}{2} - \rho(\xi)\xi^2 - \frac{1}{2}\tilde{\mu}_0^T\tilde{\Sigma}_0^{-1}\tilde{\mu}_0 + \frac{1}{2}\tilde{\mu}^T\tilde{\Sigma}^{-1}\tilde{\mu} + \frac{1}{2}\log\frac{|\tilde{\Sigma}|}{|\tilde{\Sigma}_0|}). \qquad (6)$$

3.2.2 TRUNCATED VARIATIONAL DISTRIBUTION

Another difficulty that must be addressed is the computational complexity of the infinite stick-breaking model. Following Blei and Jordan (2005), we employ a truncated stick-breaking representation for the variational distribution. It is worth noting that in the VB framework, the model is a non-truncated full Dirichlet process while the variational distribution is truncated. Empirical results on truncation level selection are given by Blei and Jordan (2005), but no theoretical criterion has been developed in the VB framework up to now. In all the experiments presented here we set the truncation level equal to the number of tasks. There is no loss of generality with this approach but it is computationally expensive if there are a large number of tasks.

Let K denote the truncation level. In the SMTL-1 model, the factorized variational distribution is specified as

$$q(\mathbf{Z}) = [\prod_{m=1}^{M} q_{c_m}(c_m)] \cdot [\prod_{k=1}^{K} q_{v_k}(v_k)] \cdot q_\alpha(\alpha) \cdot [(\prod_{k=1}^{K} q_{w_k^*}(w_k^*)],$$

where

- $q_{c_m}(c_m)$ is a multinomial distribution,

$$c_m \sim M_K(1;\phi_{m,1},\ldots,\phi_{m,K}), m = 1,\ldots,M.$$

- $q_{v_k}(v_k)$ is a Beta distribution

$$v_k \sim Be(\varphi_{1,k},\varphi_{2,k}), k = 1,\ldots,K-1.$$

  Note $q_{v_K}(v_K) = \delta_1(v_K)$.

- $q_\alpha(\alpha)$ is a Gamma distribution

$$\alpha \sim Ga(\tau_1,\tau_2).$$

- $q_{w_k^*}(w_k^*)$ is a normal distribution,

$$w_k^* \sim N_d(\theta_k,\Gamma_k), k = 1,\ldots,K.$$

Similarly, in the SMTL-2 model, the factorized variational distribution is specified as

$$q(\mathbf{Z}) = [\prod_{m=1}^{M} q_{c_m}(c_m)] \cdot [\prod_{k=1}^{K} q_{v_k}(v_k)] \cdot q_\alpha(\alpha) \cdot [(\prod_{k=1}^{K} q_{w_k^*}(w_k^*)] \cdot q_{\mu,\{\lambda_j\}_{j=1}^d}(\mu, \{\lambda_j\}_{j=1}^d).$$

where $q_{c_m}(c_m), q_{v_k}(v_k), q_\alpha(\alpha)$ and $q_{w_k^*}(w_k^*)$ are the same as the specifications above.

The distribution $q_{\mu,\{\lambda_j\}_{j=1}^d}(\mu, \{\lambda_j\}_{j=1}^d)$ is Normal-Gamma,

$$(\mu, \{\lambda_j\}_{j=1}^d) \sim N_d(\eta, (\beta\Lambda)^{-1}) \prod_{j=1}^{d} Ga(\gamma_{1j}, \gamma_{2j}). \tag{7}$$

A coordinate ascent algorithm is developed for the SMTL model by applying the mean-field method (Ghahramani and Beal, 2001). Each factor in the factorized variational distribution and ξ, the variational parameter of the sigmoid function, are re-estimated iteratively conditioning on the current estimate of all the others, assuring the lower bound of the log likelihood increases monotonically until it converges. The re-estimation equations can be found in the Appendix.

3.3 Prediction for Test Samples

The prediction function for a new test sample $x_{m,\star}$ is

$$p(y_{m,\star} = 1 | c_m, \{w_k^*\}_{k=1}^K, x_{m,\star}) = \sum_{k=1}^{K} c_{m,k} \sigma({w_k^*}^T x_{m,\star}). \tag{8}$$

Integrating (8) over the variational distributions $q_{w_k^*}(w_k^*)$ and $q_{c_m}(c_m)$ yields

$$\begin{aligned} & p(y_{m,\star} = 1 | \{\{\phi_{m,k}\}_{k=1}^K\}_{m=1}^M, \{\theta_k\}_{k=1}^K, \{\Gamma_k\}_{k=1}^K, x_{m,\star}) \\ = & \sum_{k=1}^{K} \phi_{m,k} \int \sigma({w_k^*}^T x_{m,\star}) N_d(\theta_k, \Gamma_k) dw_k^*. \end{aligned} \tag{9}$$

The integral in (9) does not have an analytic form. The variational method described in Section 3.2.1 could give us a lower bound of the integral in the form of (6), if we apply the EM algorithm by taking $N_d(\theta_k, \Gamma_k)$ as the prior of w_k^* and $(x_{m,\star}, y_{m,\star} = 1)$ as the observation. However, we prefer an accurate estimate of the integral instead of the lower bound of it. In addition, the iterative EM algorithm might be inefficient in some applications that have a strict requirement to the testing speed. Therefore, we use the approximate form of the integral in MacKay (1992)

$$\int \sigma({w_k^*}^T x_{m,\star}) N_d(\theta_k, \Gamma_k) dw_k^* \approx \sigma(\frac{\theta_k^T x_{m,\star}}{\sqrt{1 + \frac{PI}{8} x_{m,\star}^T \Gamma_k x_{m,\star}}}), \tag{10}$$

where PI is the constant approximately 3.1416 (here PI is used instead of π to avoid confusion with π used in the stick-breaking representation of the DP).

We design a simple experiment to empirically evaluate accuracy of the approximation. Assume $x_{m,\star}$ is a 1-dimensional vector. Let the value of θ_k be $-10, -9.5, \cdots, 10$ and the value of Γ_k be $10^{-1}, 10^0, 10^1$ and 10^2. For any combination of the values of θ_k and Γ_k, we compare the average error between the approximation and the true value of the integration over the range of $x_{m,\star}$ from -10 to 10 with an interval of 0.5. The "true" value of the integral is approximated by the MCMC

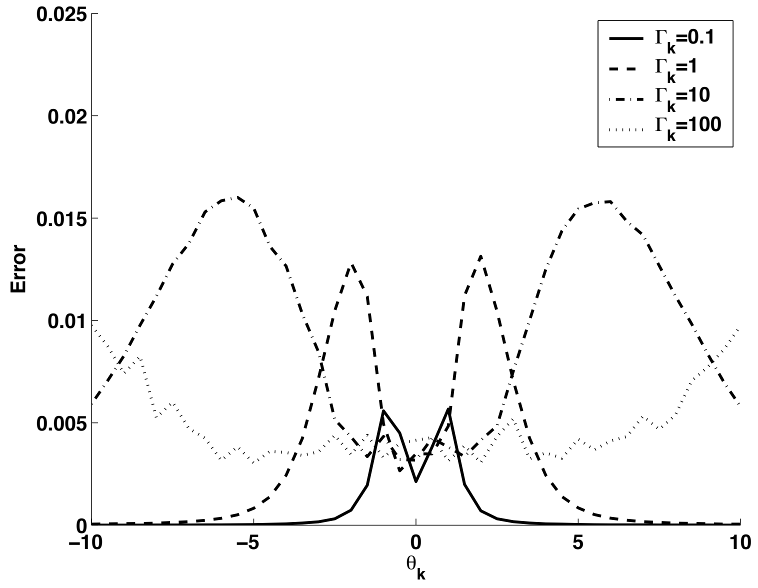


Figure 2: The error between the approximation and the true value of the integral in (10), averaged over the range of $x_{m,\star}$ from -10 to 10 with an interval of 0.5.

method, that is, we randomly draw 10^4 samples of w_k^* from the normal distribution $N_d(\theta_k, \Gamma_k)$, substitute the samples into the logistic function $\sigma(w_k^{*T} x_{m,\star})$ and take the average of the function values. The results are plotted in Fig. 2, which shows the approximation is rather accurate.

Substituting (10) into (9) yields the prediction function as

$$p(y_{m,\star} = 1 | \{\{\phi_{m,k}\}_{k=1}^K\}_{m=1}^M, \{\theta_k\}_{k=1}^K, \{\Gamma_k\}_{k=1}^K, x_{m,\star}) \approx \sum_{k=1}^K \phi_{m,k} \sigma\left(\frac{\theta_k^T x_{m,\star}}{\sqrt{1+\frac{PI}{8} x_{m,\star}^T \Gamma_k x_{m,\star}}}\right).$$

4. Information Transfer to a New Task (AMTL)

Assume the SMTL model has been applied to M tasks. Now assume a new task $M+1$ is considered. When learning task $M+1$ we wish to benefit from the information learned from the previous M tasks. One option is to re-run the SMTL algorithm on all $M+1$ tasks, but this requires storage of the data from all previous tasks and may be computationally prohibitive for real-time applications. In fact, re-running SMTL on all tasks is in many situations not necessary, as the previous M tasks may simply represent the history and re-estimating them is uninteresting. In these cases, we need only concentrate on learning the new task, treating the previous M tasks as the background tasks, from which relevant information is transferred to the new task. Such an approach provides us the advantage of algorithmic efficiency, since we do not have to manipulate a bulk of data from the past tasks. In this section, we develop an efficient method for fast learning of the new observed task. Since task $M+1$ is treated differently from tasks 1 though M, this is referred to as asymmetric MTL, or AMTL.

4.1 Prior Learned from Previous Tasks

From (2), we know that the conditional distribution of the classifier w_{M+1} given α, G_0 and the other M classifiers is

$$p(w_{M+1}|w_1,\cdots,w_M,\alpha,G_0) = \frac{\alpha}{M+\alpha}G_0 + \frac{1}{M+\alpha}\sum_{k=1}^{K} n_k \delta_{w_k^*}, \tag{11}$$

where $n_k = \sum_{m=1}^{M} c_{m,k}$ is the number of w_m which are equal to w_k^*.

Assume the SMTL-1 model has been applied to M previous tasks and thus the variational distributions $q_{c_m}(c_m)$, $q_\alpha(\alpha)$ and $q_{w_k^*}(w_k^*)$ have been optimized. We have $E_q[c_{m,k}] = \phi_{m,k}$, $E_q[\alpha] = \frac{\tau_1}{\tau_2}$ and $E_q[w_k^*] = \theta_k$, where E_q is the expectation with respect to the variational distributions. Substituting the expectations into (11) yields

$$p(w_{M+1}|\{\{\phi_{m,k}\}_{k=1}^{K}\}_{m=1}^{M},\{\theta_k\}_{k=1}^{K},\tau_1,\tau_2,G_0) \approx \frac{\frac{\tau_1}{\tau_2}}{M+\frac{\tau_1}{\tau_2}}G_0 + \frac{1}{M+\frac{\tau_1}{\tau_2}}\sum_{k=1}^{K}\bar{n}_k\delta_{\theta_k}, \tag{12}$$

where $\bar{n}_k = \sum_{m=1}^{M}\phi_{m,k}$. For future convenience, we define $\Omega = \{\{\{\phi_{m,k}\}_{k=1}^{K}\}_{m=1}^{M},\{\theta_k\}_{k=1}^{K},\tau_1,\tau_2\}$.

Equation (12) represents our belief about the classifier w_{M+1} before we actually see the data $\mathcal{D}_{M+1}$. Therefore, by taking it as a prior for w_{M+1}, information learned from previous tasks can be transferred to the new task. The posterior of w_{M+1}, after observing the data $\mathcal{D}_{M+1}$, is computed according to the Bayes Theorem

$$p(w_{M+1}|\mathcal{D}_{M+1},\Omega,G_0) = \frac{p(\mathcal{D}_{M+1}|w_{M+1})p(w_{M+1}|\Omega,G_0)}{p(\mathcal{D}_{M+1}|\Omega,G_0)}, \tag{13}$$

where

$$\begin{aligned}
&p(\mathcal{D}_{M+1}|w_{M+1})\\
=\;& \prod_{n=1}^{N_{M+1}} p(y_{M+1,n}|w_{M+1},x_{M+1,n}),\\
=\;& \prod_{n=1}^{N_{M+1}} \sigma(w_{M+1}^T x_{M+1,n})^{y_{M+1,n}}[1-\sigma(w_{M+1}^T x_{M+1,n})]^{1-y_{M+1,n}}.
\end{aligned} \tag{14}$$

Note in Section 4 we limit the discussion on learning from previous tasks to the SMTL-1 model, for which the parameters of G_0 are given; the approach developed in this section can be extended to the SMTL-2 model by substituting the expectations on the parameters of G_0 into (12).

4.2 Sampling Posterior Using Metropolis-Hastings Algorithm

The posterior (13) does not have an analytic form because the prior (12) is a mixture of the base distribution G_0 with several point mass distributions, and the sigmoid function in the likelihood function (14) is not conjugate to the prior. Considering that the computational burden is small in the AMTL case (we only deal with data from task $M+1$), we appeal to MCMC methods and develop a Metropolis-Hastings algorithm to draw samples from the posterior (Robert and Casella, 2004; Neal, 1998). This algorithm is feasible in practice since it is a simple MCMC solution and the computational cost is low.

Metropolis-Hastings Algorithm

1. Draw a sample $\dot{w}$ from (12).
2. Draw a candidate $\hat{w}$ also from (12).
3. Compute the acceptance probability
   $a(\hat{w},\dot{w}) = \min[1, \frac{p(\mathcal{D}_{M+1}|\hat{w})}{p(\mathcal{D}_{M+1}|\dot{w})}]$.
4. Set the new value of $\dot{w}$ to $\hat{w}$ with this probability; otherwise let the new value of $\dot{w}$ be the same as the old value.
5. Repeat Step 2-4 until the required number of samples, denoted by N_{SAM}, are taken.

This yields an approximation to the posterior in (13)

$$p(w_{M+1}|\mathcal{D}_{M+1},\Omega,G_0) \approx \frac{1}{N_{SAM}} \sum_{i=1}^{N_{SAM}} \delta_{\dot{w}_i}.$$

4.3 Prediction Algorithm for Test Samples in New Task

Our goal is to learn w_{M+1} so that the resulting classifier can accurately predict the class label for a new test sample $x_{M+1,\star}$. The prediction function is

$$\begin{aligned}
& p(y_{M+1,\star}=1|x_{M+1,\star},\Omega,G_0) \\
= & \int p(y_{M+1,\star}=1|x_{M+1,\star},w_{M+1})p(w_{M+1}|\mathcal{D}_{M+1},\Omega,G_0)dw_{M+1}, \\
\approx & \tfrac{1}{N_{SAM}} \sum_{i=1}^{N_{SAM}} p(y_{M+1,\star}=1|x_{M+1,\star},\dot{w}_i), \\
= & \tfrac{1}{N_{SAM}} \sum_{i=1}^{N_{SAM}} \sigma\left(\dot{w}_i^T x_{M+1,\star}\right).
\end{aligned} \tag{15}$$

Hence, the entire learning procedure for a new task is
AMTL-1 Algorithm Given Ω and G_0,

1. Compute the parameters in (12) - the prior learned from previous M tasks.
2. Draw samples $\dot{w}_1,\cdots,\dot{w}_{N_{SAM}}$ with the Metropolis-Hastings algorithm.
3. Predict for test samples using (15).

4.4 Comparison with Method of Yu et al. (2004)

Yu et al. (2004) present a hierarchical Bayesian framework for information filtering. Their purpose is to find the right information item for an active user, utilizing both item content information and an accumulated database of item ratings cast by a large set of users. The problem is modeled as a classification problem by labeling the items a user likes "1" and "0" otherwise. The learning situation is similar to our AMTL case, if each user is treated as a task. In Yu's approach, a Dirichlet process prior is learned from the database

$$p(w_{M+1}|\{D_m\}_{m=1}^M,\alpha_0,G_0) = \frac{\alpha_0}{M+\alpha_0}G_0 + \frac{1}{M+\alpha_0}\sum_{m=1}^M \zeta_m \delta_{\hat{w}_m}, \tag{16}$$

where α_0 and G_0 are pre-defined. They treat $\hat{w}_m$ as the maximum a posteriori (MAP) estimate of task m. The weights ζ_m are learned with an Expectation Maximization (EM) algorithm presented in Yu et al. (2004, Section 3).

From the stick-breaking view of DP, the prior in (16) is an approximation to the standard DP, in that the locations of sticks are fixed, at the classifiers learned from individual models, while the weights are inferred with the EM algorithm. This approximation is inappropriate if the training samples in each previous task are not sufficient, so that each task cannot learn an accurate classifier by only using the corresponding user's profile. In other words, it is not a good use of the information in previous tasks.

Some other approximations are made in the prediction step of Yu's approach, although they are relatively trivial compared to the approximation to the DP prior mentioned above. For comparison, we develop the second AMTL algorithm with the approximate DP prior.

AMTL-2 Algorithm Given $\hat{w}_1, \cdots, \hat{w}_M$, α_0 and G_0,

1. Optimize the weights ζ_m in (16) with the EM algorithm in Yu et al. (2004, Section 3).

2. Substitute (12) with (16), then draw samples $\dot{w}_1, \cdots, \dot{w}_{N_{SAM}}$ with the Metropolis-Hastings algorithm.

3. Predict for test samples using (15).

Empirical comparisons of the two AMTL algorithms are reported in Section 5.1.2.

5. Experiments and Results Analysis

An empirical study of the proposed methods is conducted on two real applications: (i) a landmine detection problem, and (ii) an art image retrieval problem.

5.1 Landmine Detection

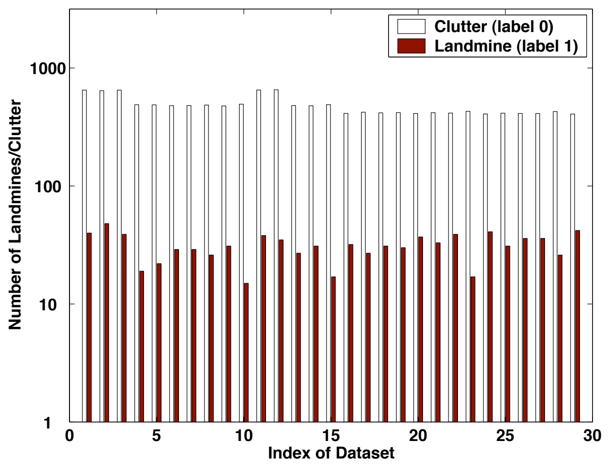


Figure 3: Number of landmines/clutter in each of the 29 data sets.

Data from 29 tasks are collected from various landmine fields.[2] Each object in a given data set is represented by a 9-dimensional feature vector and the corresponding binary label (1 for landmine and 0 for clutter). The feature vectors are extracted from radar images, concatenating four moment-based features, three correlation-based features, one energy ratio feature and one spatial variance feature. Figure 3 shows the number of landmines/clutter in each data set.

The landmine detection problem is modeled as a binary classification problem. The objective is to learn a classifier from the labeled data, with the goal of providing an accurate prediction for an unlabeled feature vector. We treat classification of each data set as a learning task and evaluate the proposed SMTL and AMTL methods on this landmine detection problem.

Among these 29 data sets, 1-15 correspond to regions that are relatively highly foliated and 16-29 correspond to regions that are bare earth or desert. Thus we expect that there are approximately two clusters of tasks corresponding to two classes of ground surface conditions. We first evaluate the SMTL models using data sets 1-10 and 16-24; next, for the AMTL setting, these data are treated as previous tasks and data sets 11-15 and 25-29 are treated as new observed tasks.

5.1.1 SMTL

Data sets 1-10 and 16-24 are used for the SMTL experiment, so there are a total of 19 tasks. We examine the performance of four methods on accuracy of label prediction: (i) SMTL-1, (ii) SMTL-2, (iii) the STL method—learn each classifier using the corresponding data set only—with the variational approach to logistic regression models in Section 3.2.1, and (iv) simply pooling the data in all tasks and then learning a single classifier with the variational approach as for (iii).

The performance is measured by average AUC on 19 tasks, where AUC denotes area under the Receiver Operation Characteristic (ROC) curve. A larger AUC value indicates a better classification performance. To have a comprehensive evaluation, we test the algorithms with different sizes of training sets. The number of training samples for every task is set as $20, 40, \cdots, 300$. For each task, the training samples are randomly chosen from the corresponding data set and the remaining samples are used for testing. Since the data have severely unbalanced labels, as shown in Fig. 3, we have a special setting that assures there is at least one "1" and one "0" sample in the training set of each task.

Hyper-parameter settings are as follows: (i) $\tau_{10} = 5e^{-2}, \tau_{20} = 5e^{-2}, \gamma_{10} = 1e^{-2}, \gamma_{20} = 1e^{-3}$ and $\beta_0 = 1e^{-2}$, (ii) $\tau_{10} = 5e^{-2}, \tau_{20} = 5e^{-2}, \mu = \mathbf{0}$ and $\Sigma = 10\mathbf{I}$, (iii) and (iv) $\tilde{\mu}_0 = \mathbf{0}$ and $\tilde{\Sigma}_0 = 10\mathbf{I}$. We also tested with other choices of hyper-parameters and found that the algorithms are not sensitive to the hyper-parameter settings as long as the hyper-priors are rather diffuse.

We plot the results of 100 random runs in Fig. 4. The two SMTL methods generally outperform the STL method and simple pooling. To gain insight into how the SMTL method identifies the clustering structure of tasks, we calculate the between-task similarity matrix as follows: at a certain setting of the size of training set (e.g., 20 training samples per task), for each random run, the SMTL algorithm outputs the variational distribution $q_{c_m}(c_m)$ with the optimized variational parameters $\{\phi_{m,k}\}_{k=1}^{K}$, where $\phi_{m,k}$ indicates the probability that task m belongs to cluster k, and then we take $k_m^* = \arg\max_k \phi_{m,k}$ as the membership of task m. The element at the ith row and jth column in the between-task similarity matrix records number of occurrences, among 100 random runs, that task i and task j are grouped into the same cluster. Fig. 5 shows the Hinton diagram (Hinton and Sejnowski, 1986) for the between-task similarity matrices corresponding different experiment

2. The data are available at http://www.ee.duke.edu/∼lcarin/LandmineData.zip.

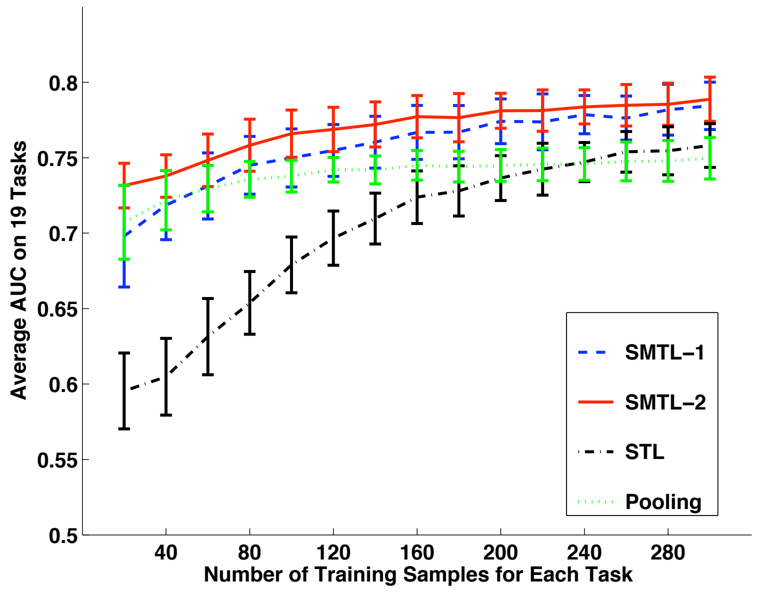


Figure 4: Average AUC on 19 tasks in the landmine detection problem.

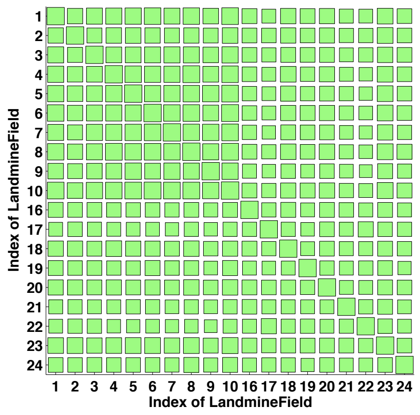
settings: (a)(c)(e) learned by using the SMTL-1 model with 20, 100 and 300 training samples per task; (b)(d)(f) learned by using the SMTL-2 model with 20, 100 and 300 training samples per task. In a Hinton diagram, the size of blocks is proportional to the value of the corresponding matrix elements.

With the help of Fig. 5, we have the following analysis on the behavior of the curves in Fig. 4.

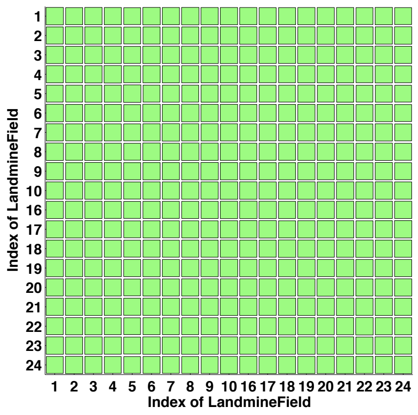
1. When very few training samples are available, for example, 20 per task, the STL method performs poorly. The simple pooling method significantly improves the performance because its effective training size is 18 times larger than that for each individual task. The training samples are so few that although both SMTL methods find that all tasks are similar, they cannot identify the extent of similarity between tasks (see (a) and (b) in Fig. 5). As a result, they perform similar to the simple pooling method. The SMTL-2 performs slightly better due to additional robustness introduced by integrating over the parameters of G_0.

2. When there are a few training samples available, for example, 100 per task, the simple pooling method does not improve further as more training samples are pooled together, because it ignores the statistical differences between tasks. Both SMTL methods begin to learn the clustering structure (see (c) and (d) in Fig. 5) and this leads to better performance than the simple pooling. The clustering structure in (d) is more obvious than that in (c), therefore the SMTL-2 method works slightly better than the SMTL-1 method.

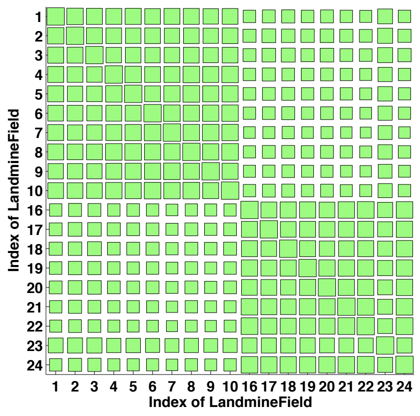
3. When each task has many training samples, for example, 300 per task, both SMTL methods identify the clustering structure (see (e) and (f) in Fig. 5). The number of training samples is large enough for each task to learn well by itself, so the curve for the STL method approaches the curves for the SMTL methods and exceeds the curve for the simple pooling. It is clear



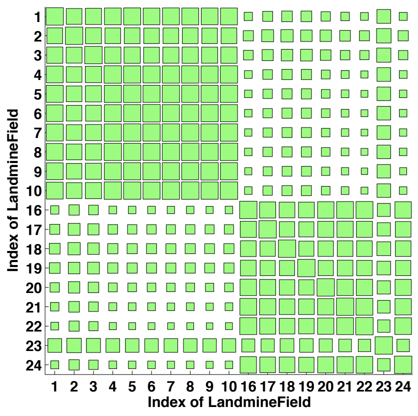
(a)



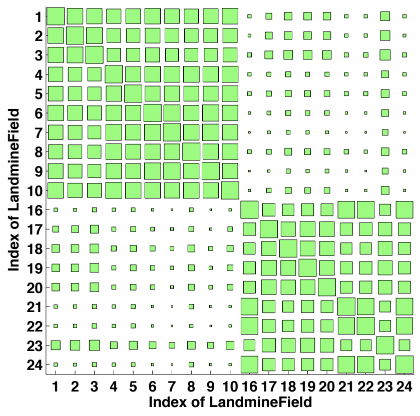
(b)



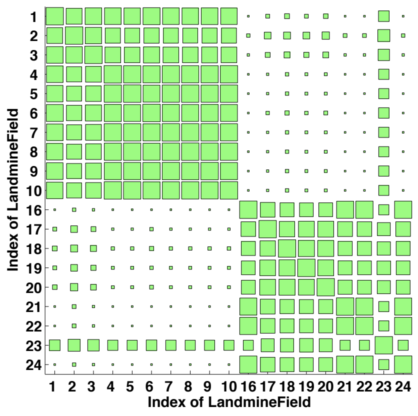
(c)



(d)



(e)



(f)

Figure 5: Hinton diagram for the between-task similarity matrix in the landmine detection problem, (a)(c)(e) learned by using the SMTL-1 model with 20, 100 and 300 training samples per task; (b)(d)(f) learned by using the SMTL-2 model with 20, 100 and 300 training samples per task. In a Hinton diagram, the size of blocks is proportional to the value of the corresponding matrix elements.

in (e) and (f) that the 19 tasks are roughly grouped into two clusters, which agree with the ground truth discussed above.

5.1.2 AMTL

In this experiment, the data sets used in Section 5.1.1 are treated as previous tasks and data sets 11-15 and 25-29 as the new observed tasks. We compare three approaches:

1. AMTL-1: First we apply the SMTL-1 model to those previous tasks and learn the variational parameters Ω, and then run the AMTL-1 algorithm on each new task.

2. AMTL-2: First we apply the variational logistic regression approach in Section 3.2.1 to each previous task and learn the individual classifiers $\hat{w}_1, \ldots, \hat{w}_M$, and then run the AMTL-2 algorithm on each new task.

3. STL: Each new task learns by itself.

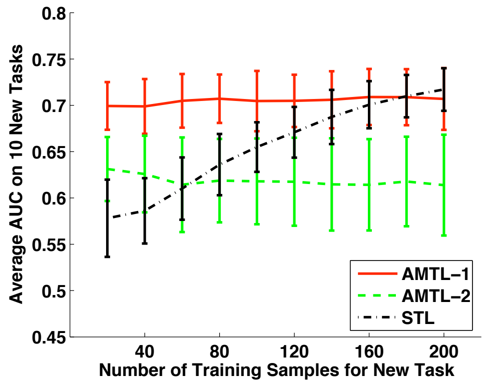
In the two AMTL methods, the mean and variance of the base distribution G_0 are specified as $\mu = \mathbf{0}$ and $\Sigma = 10\mathbf{I}$. The parameters τ_1 and τ_2 in the AMTL-1 model can be estimated from previous tasks, while α_0 in the AMTL-2 model is a predefined parameter, which represents a prior belief about the relatedness between the new task and previous tasks. We have two settings: (i) $\alpha_0 = \frac{\tau_1}{\tau_2} = 0$, which represents a belief that the new task is closely related to previous tasks, and (ii) $\alpha_0 = \frac{\tau_1}{\tau_2}$, where τ_1 and τ_2 are estimated from previous tasks.

The performance of the three approaches is measured by the average AUC over the 10 new tasks. Experimental results of 100 random runs are shown in Fig. 6. Two factors affecting learning performance are considered: (i) number of training samples per *previous* task, used for learning the prior of w_{M+1}, and (ii) number of training samples per *new* task. The first factor is evaluated at 40, 160 and all samples in each *previous* task used for training, corresponding to (a)(b), (c)(d) and (e)(f) in Fig. 6 respectively. The second factor is evaluated at $20, 40, \cdots, 200$ training samples in each *new* task, plotted along the horizonal axis in each subplot of Fig. 6.

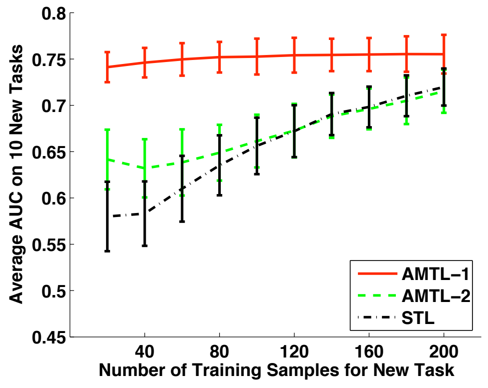
We have the following observations from Fig. 6:

1. In the case $\alpha_0 = \frac{\tau_1}{\tau_2} = 0$ (see (a)(c)(e)), the prior belief is that the new task is closely related previous tasks and the purpose of this setting is to focus on the comparison of information transferability of the two AMTL approaches, given the ground truth that the *new* tasks are quite similar to some of *previous* tasks. The AMTL-1 approach efficiently transfers information from *previous* tasks to the *new* task. The SMTL method can learn an informative prior for the *new* task, with only 40 training samples for each *previous* task (see (a) in Fig. 6). The learning performance is slightly improved by using more training samples for each *previous* task. The performance has almost no change as the number of training samples in the *new* task increases, because we use the linear classier and thus the matching classifier can be found with even only a pair of "1" and "0" labeled samples.

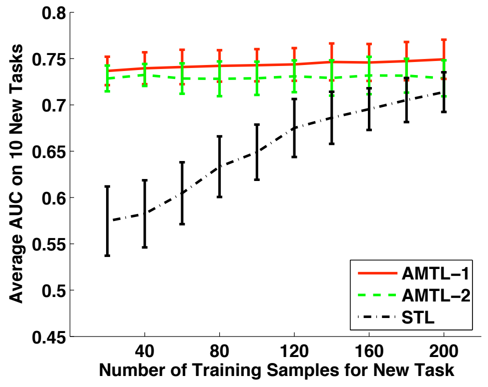
   Information transferability of the AMTL-2 approach is weaker than the AMTL-1 approach, due to the approximate DP prior (as analyzed in Section 4.4). As a result, the more training samples in the *new* task, the more confused the algorithm is about which "stick" should be the matching classifier. This explains why the curve for the AMTL-2 approach in (a) even drops a little as the training samples in the *new* task increases. However, with a large set



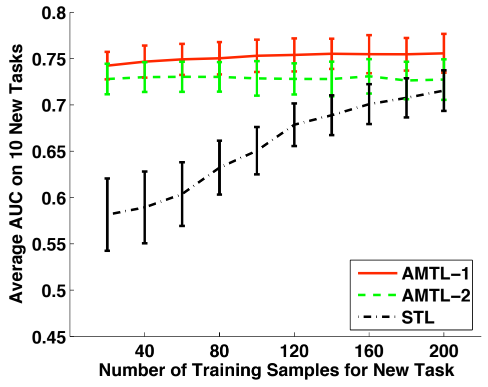
(a) $\alpha_0 = \frac{\tau_1}{\tau_2} = 0$; 40 training samples for each *previous* task.



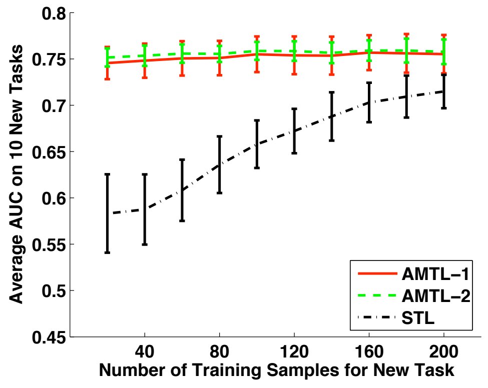
(b) $\alpha_0 = \frac{\tau_1}{\tau_2}$, where τ_1 and τ_2 are estimated from previous tasks; 40 training samples for each *previous* task.



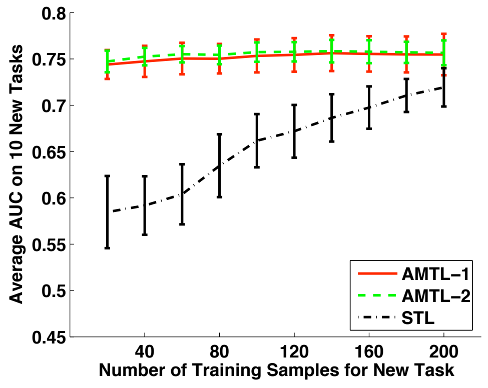
(c) $\alpha_0 = \frac{\tau_1}{\tau_2} = 0$; 160 training samples for each *previous* task.



(d) $\alpha_0 = \frac{\tau_1}{\tau_2}$, where τ_1 and τ_2 are estimated from previous tasks; 160 training samples for each *previous* task.



(e) $\alpha_0 = \frac{\tau_1}{\tau_2} = 0$; all samples in *previous* tasks used for training.



(f) $\alpha_0 = \frac{\tau_1}{\tau_2}$, where τ_1 and τ_2 are estimated from previous tasks; all samples in *previous* tasks used for training.

Figure 6: Average AUC on 10 new tasks in the landmine detection problem.

of training samples in *previous* tasks, the AMTL-2 approach works slightly better than the AMTL-1 approach (see (e)), because the former can tell the subtle difference between very similar tasks, while the latter treats them as identical.

2. Next, we recover G_0 by using the parameters τ_1 and τ_2 estimated from *previous* tasks in the AMTL-1 model and setting $\alpha_0 = \frac{\tau_1}{\tau_2}$ in the AMTL-2 model (see (b)(d)(f)). That enables the new task to discover a new classifier by itself as well as use those learned from previous tasks. Information transferability of the AMTL-2 approach is weak when there are only a few training samples for each *previous* task, as disussed above. In such a case, the AMTL-2 approach works just as the STL approach because the *new* task has to learn by itself (see (b)). In contrast, the AMTL-1 approach benefits from *previous* tasks so that incorporation of G_0 has a relatively small effect on its performance.

5.2 Art Image Retrieval

A web survey is built to collect user ratings on 642 paintings from 30 artists.[3] A user chooses his rating of an image from “like”,“dislike” or “not sure”. Every user may give ratings only on a subset of all images. In total 203 user ratings are collected.

Our objective is to estimate a user’s preference on unrated images. We model this as a binary classification problem. Each user corresponds to a classification task. The images he rates as “like” are labeled “1”, “dislike” labeled “0” and the images with the rating “not sure” are not included. The content of an image is described by a 275-dimensional (275-D) feature vector concatenating a 256-D correlagram, a 10-D Pyramid wavelet texture and 9-D first and second color moments.

The painting image data differ with the landmine data in two respects. First, the low-level features of image content, for example, color and texture, are weak indicators of human preferences, therefore the content of an image is less helpful than ratings on that image from other users with similar interests. Second, because user preferences are very diverse, the clustering structure of tasks is expected to be more complex than that of the landmine detection tasks.

We use the 68 users who rate more than 100 images for the SMTL experiment. Then we take these as previous tasks and those 50 users who rate between 50 and 100 images are treated as new tasks in the AMTL experiment.

5.2.1 SMTL

In the SMTL experiment, two methods are compared: (i) SMTL-1, and (ii) the single-task learning (STL) method using the variational logistic regression approach in Section 3.2.1. The simple pooling method is not feasible because different users may look at the same image and give different ratings. The SMTL-2 model is also excluded, because the feature dimension (275) is high relative to the number of training samples for each task, so that it is hard to get an accurate estimation on the variational distribution of λ_j in (7), which is the precision on each feature dimension.

Hyper-parameter settings are as follows: (i) $\tau_{10} = 5e^{-2}$, $\tau_{20} = 5e^{-2}$, $\mu = \mathbf{0}$ and $\Sigma = 10\mathbf{I}$, and (ii) $\tilde{\mu}_0 = \mathbf{0}$ and $\tilde{\Sigma}_0 = 10\mathbf{I}$. Similar to the landmine experiments, the performance is measured by the average AUC on all tasks and evaluated at 10, 20, 30, 40 or 50 randomly selected training samples for each task. Figure 7 plots the results of 10 random runs.

3. The survey is online at http://honolulu.dbs.informatik.uni-muenchen.de:8080/paintings/index.jsp.

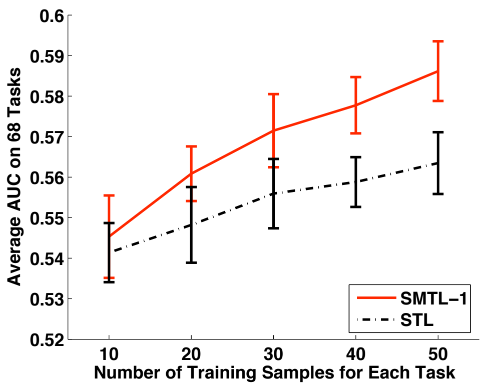


Figure 7: Average AUC on 68 tasks in the art image retrieval problem.

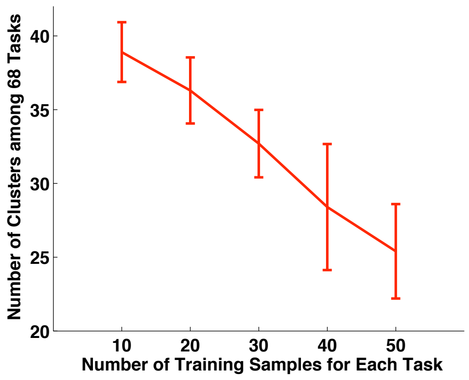


Figure 8: Number of clusters among 68 tasks in the art image retrieval problem.

As mentioned above, the tasks in the art image database are more diverse than those in the landmine data sets. To get a clear view of this, we observe the number of clusters among 68 tasks instead of the between-task similarity matrix. As in Section 5.1.1, we make a hard decision on membership of each task, for each evaluation point and each random run, and then we obtain the statistics of number of clusters among 68 tasks, which is shown in Fig. 8. When the training size is small, the algorithm weakly finds the similarity between tasks and most of the tasks learn by themselves, therefore the SMTL-1 method works similar to the single-task learning. As the number of training samples increases, the clustering structure becomes more clear and information is shared between the users/tasks with similar interests, leading to improvement in learning performance.

5.2.2 AMTL

We first apply the SMTL-1 method on the 68 tasks, using all data as training samples, to learn the prior for a new classifier, then evaluate the performance of the AMTL-1 algorithm on 50 new tasks, measured by the average AUC. The performance is compared to that of the STL approach, which means learning by the new task itself. The number of samples drawn with the Metropolis-Hastings

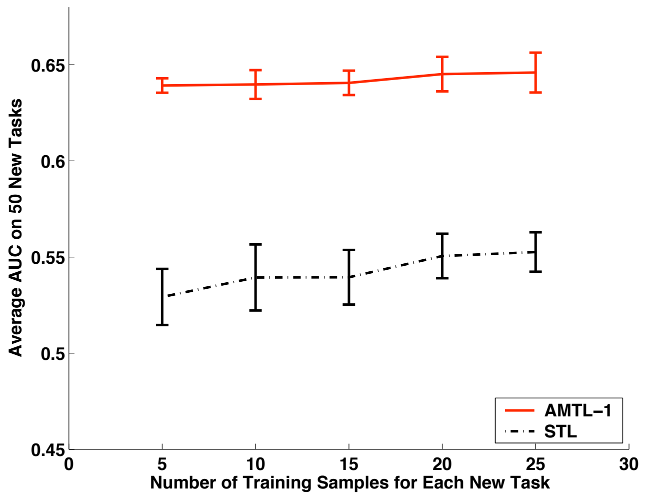


Figure 9: Average AUC on 50 new tasks in the art image retrieval problem.

algorithm in Section 4.2, N_{SAM}, is set to be $1e^3$. The results of 10 random runs are shown in Fig. 9. The curves indicate that the AMTL-1 approach outperforms the STL approach.

6. Conclusions and Future Work

A DP-based multi-task learning algorithm has been applied to the problem of designing logistic-regression classifiers for multiple tasks, for cases in which there is the potential of enhancing individual-task performance via appropriate sharing of inter-task data. Two overarching formulations have been considered. In the symmetric multi-task learning (SMTL) formulation all of the task-dependent classifiers are learned jointly. While this is a useful formulation in many cases, it requires one to store all data across previous tasks. In many cases we may undertake a new task and we would like this task to benefit from experience acquired from previous tasks, without having to return to all data observed previously. This has motivated what we have termed an asymmetric multi-task learning (AMTL) formulation. In addition to the overarching SMTL and AMTL formulations, we have considered different forms of these algorithms based on how the DP priors are handled.

MTL classification performance has been presented on two data sets: (i) a landmine sensing problem based on measured data, and (ii) an art-preference database. Concerning (i), the MTL formulation yielded a clear indication of how the data from the multiple tasks clustered into related physical phenomena. For this data we know the task-dependent environmental conditions under which the sensing was performed, and the task relatedness reflected in Hinton maps demonstrated close agreement with physical expectations. This provides a powerful confirmation of the utility of the DP formulation for a case in which "truth" is known, yielding confidence for new multi-task data sets for which the DP formulation may be used to infer truth.

In the context of the DP formulation, we considered examples for which the innovation parameter and the parameters of the base distribution were fixed, while in a separate formulation prior distributions were placed on these parameters (yielding a further layer in the Bayesian formulation). For the examples presented here we found the performance of the latter formulation to be slightly better than the former. We also compared the DP MTL formulation to several simpler learning approaches: (i) single-task learning in which no data are shared, (ii) pooling in which all data are shared indiscriminately, and (iii) a simplification to DP developed by Yu et al. (2004). For the data considered, the DP-based MTL formulations developed here outperformed these simpler approaches.

In future research we plan to extend the MTL approach to more-general settings. For example, in the MTL classifier formulation it is assumed that labeled data are available for each of the tasks. More realistically, labeled data may be available from previous tasks, but the new task under investigation may only contain unlabeled data. Based on examining the relationship of the data manifold of the new unlabeled data relative to the manifolds of the previous tasks, it may be possible to ascertain which labeled data, from previous tasks, are relevant for the new unlabeled data under test. In such a heterogeneous MTL setting, involving tasks characterized by labeled and unlabeled data, it may be possible to label the new data under test (from the new task) without requiring any associated labeled data.

In this context one may also consider an active-learning setting, in which labels are acquired selectively from the new task of interest, such that after active learning all tasks have labeled data and the MTL formulations presented here may be applied directly. In this context we note that such an approach was examined in the course of the research presented here, with active learning performed using query by committee (QBC) (McCallum and Nigam, 1998). For the data considered in this paper, we found that as long as at least one label was acquired from each of the two labels (we here considered binary labels), the MTL algorithm performed well. Consequently, while the QBC results were good, even a random acquisition of labels yielded good MTL performance, as long as at least one (randomly acquired) label existed from each of the two labels. This suggests a significant robustness of the MTL formulation, in its ability to use a small amount of labeled data for a given task to still yield good task-dependent classification performance, by appropriately sharing labeled data from other tasks. Nevertheless, this phenomenon is worth further examination, with other data sets, to further examine the utility of active learning as applied to unlabeled data from a new task (relative to simply using random sampling to determine which data to acquire labels on).

Acknowledgments

We would like to acknowledge Kai Yu and Volker Tresp for providing the art image data.

Appendix A. Re-Estimation Equations in Coordinate Ascent Algorithm

For the SMTL-1 model, the re-estimation equations are as follows

- $q_{c_m}(c_m)$:

$$\begin{aligned} s_{m,k} \;=\; & \sum_{n=1}^{N_m}[-\rho(\xi_{m,n})x_{m,n}^T(\theta_k\theta_k^T+\Gamma_k)x_{m,n}+(y_{m,n}-\tfrac{1}{2})\theta_k^T x_{m,n} \\ & +\log(\sigma(\xi_{m,n}))-\tfrac{1}{2}\xi_{m,n}+\rho(\xi_{m,n})\xi_{m,n}^2] \\ & +\mathbf{1}(k<K)[\Psi(\varphi_{1,k})-\Psi(\varphi_{1,k}+\varphi_{2,k})] \\ & +\mathbf{1}(k>1)\{\sum_{i=1}^{k-1}[\Psi(\varphi_{2,i})-\Psi(\varphi_{1,i}+\varphi_{2,i})]\} \end{aligned},$$

$$\phi_{m,k}=\frac{\exp(s_{m,k})}{\sum_{k=1}^{K}\exp(s_{m,k})},$$

$$m=1,\ldots,M,k=1,\ldots,K,$$

where $\Psi(x)=\frac{d\ln\Gamma(x)}{dx}$; $\Gamma(x)$ is the Gamma function; $\mathbf{1}(E)$ is equal to 1 if the logic expression E is true and 0 otherwise.

- $q_{v_k}(v_k)$:

$$\begin{aligned} &\varphi_{1,k}=1+\sum_{m=1}^{M}\phi_{m,k}, \\ &\varphi_{2,k}=\frac{\tau_1}{\tau_2}+\sum_{m=1}^{M}\sum_{i=k+1}^{K}\phi_{m,i}, \\ &k=1,\ldots,K-1. \end{aligned}$$

- $q(\alpha)$:

$$\begin{aligned} &\tau_1=\tau_{10}+K-1, \\ &\tau_2=\tau_{20}-\sum_{k=1}^{K-1}[\Psi(\varphi_{2,k})-\Psi(\varphi_{1,k}+\varphi_{2,k})]. \end{aligned}$$

- $q_{w_k^*}(w_k^*)$:

$$\begin{aligned} &\Gamma_k=[\Sigma^{-1}+2\sum_{m=1}^{M}\phi_{m,k}\sum_{n=1}^{N_m}|\rho(\xi_{m,n})|x_{m,n}x_{m,n}^T]^{-1}, \\ &\theta_k=\Gamma_k[\Sigma^{-1}\mu+\sum_{m=1}^{M}\phi_{m,k}\sum_{n=1}^{N_m}(y_{m,n}-\frac{1}{2})x_{m,n}], \\ &k=1,\ldots,K. \end{aligned} \tag{17}$$

- $\xi_{m,n}$:

$$\begin{aligned} &\xi_{m,n}=\sqrt{\sum_{k=1}^{K}\phi_{m,k}x_{m,n}^T(\theta_k\theta_k^T+\Gamma_k)x_{m,n}}, \\ &m=1,\ldots,M,n=1,\ldots,N_m. \end{aligned}$$

For the SMTL-2 model, we only need to modify (17) and add an update step for μ and $\lambda_1, \cdots, \lambda_d$

- $q_{w_k^*}(w_k^*)$:

$$\begin{aligned}
\Gamma_k &= [\Delta + 2\sum_{m=1}^{M} \phi_{m,k} \sum_{n=1}^{N_m} |\rho(\xi_{m,n})| x_{m,n} x_{m,n}^T]^{-1}, \\
\theta_k &= \Gamma_k [\Delta\eta + \sum_{m=1}^{M} \phi_{m,k} \sum_{n=1}^{N_m} (y_{m,n} - \frac{1}{2}) x_{m,n}], \\
k &= 1, \ldots, K,
\end{aligned}$$

where Δ is a diagonal matrix with diagonal elements $\frac{\gamma_{1,1}}{\gamma_{2,1}}, \ldots, \frac{\gamma_{1,d}}{\gamma_{2,d}}$.

- $q_{\mu,\{\lambda_j\}_{j=1}^d}(\mu, \{\lambda_j\}_{j=1}^d)$:

$$\begin{aligned}
\beta &= \beta_0 + K, \\
\eta &= \frac{\sum_{k=1}^{K} \theta_k}{\beta}, \\
\gamma_{1,j} &= \gamma_{10} + \frac{K}{2}, \\
\gamma_{2,j} &= \gamma_{20} + \frac{1}{2}\sum_{k=1}^{K} (\theta_{k,j}^2 + \Gamma_{k,j}) - \frac{1}{2}\beta\eta_j^2, \\
j &= 1, \ldots, d,
\end{aligned}$$

where $\theta_{k,j}$ denotes the jth element of the vector θ_k (same for η_j), and $\Gamma_{k,j}$ denotes the jth diagonal element of the matrix Γ_k.

References

R.K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817C1853, 2005.

C.E. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *Annals of Statistics*, 2:1152–1174, 1974.

B. Bakker and T. Heskes. Task clustering and gating for Bayesian multitask learning. *Journal of Machine Learning Research*, 4:83–99, 2003.

J. Baxter. Learning internal representations. In *COLT: Proceedings of the Workshop on Computational Learning Theory*, 1995.

J. Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 2000.

D. Blei and M.I. Jordan. Variational inference for Dirichlet process mixtures. *Journal of Bayesian Analysis*, 1(1):121–144, 2005.

D. Burr and H. Doss. A bayesian semiparametric model for random-effects meta-analysis. *Journal of the American Statistical Association*, 100(469):242–251, Mar. 2005.

R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

F. Dominici, G. Parmigiani, R. Wolpert, and K. Reckhow. Combining information from related regressions. *Journal of Agricultural, Biological, and Environmental Statistics*, 2(3):294–312, 1997. Good literature review on the application of hierarchical models to meta analysis, Page 4.

M.D. Escobar and M. West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90:577–588, 1995.

T. Evgeniou, C.A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

T. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1: 209–230, 1973.

Z. Ghahramani and M. Beal. Propagation algorithms for variational Bayesian learning. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*. MIT Press, Cambridge, MA, 2001.

G. V. Glass. Primary, secondary and meta-analysis of research. *Educatinal Researcher*, 5, 1976.

G.E. Hinton and T.J. Sejnowski. Learning and relearning in Boltzmann machines. In McClelland J.L. Rumelhart D.E. and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1, chapter 7, pages 282–317. MIT Press, Cambridge, MA, 1986.

P.D. Hoff. Nonparametric modeling of hierarchically exchangeable data. Technical Report 421, University of Washington Statistics Department, 2003.

H. Ishwaran. Inference for the random effects in Bayesian generalized linear mixed models. In *ASA Proceedings of the Bayesian Statistical Science Section*, pages 1–10, 2000.

H. Ishwaran and L.F. James. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96:161–173, 2001.

T.S. Jaakkola and M.I. Jordan. A variational approach to Bayesian logistic regression models and their extensions. In *Proceedings of the Sixth International Workshop on Artificial Intelligence and Statistics*, 1997.

M.I. Jordan, Z. Ghahramani, T.S. Jaakkola, and L.K. Saul. An introduction to variational methods for graphical models. In M.I. Jordan, editor, *Learning in Graphical Models*. MIT Press, Cambridge, 1999.

N.D. Lawrence and J.C. Platt. Learning to learn with the informative vector machine. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

D.J.C. MacKay. The evidence framework applied to classification networks. *Neural Computation*, 4(5):720–736, 1992.

B.K. Mallick and S.G. Walker. Combining information from several experiments with nonparametric priors. *Biometrika*, 84(3):697–706, 1997.

A. McCallum and K. Nigam. Employing EM in pool-based active learning for text classification. In *Proceedings of the 15th International Conference on Machine Learning*, 1998.

S. Mukhopadhyay and A.E. Gelfand. Dirichlet process mixed generalized linear models. *Journal of the American Statistical Association*, 92(438):633–639, 1997.

P. Müller, F. Quintana, and G. Rosner. A method for combining inference across related nonparametric Bayesian models. *Journal of the Royal Statistical Society Series B*, 66(3):735–749, 2004.

R.M. Neal. Markov chain sampling methods for Dirichlet process mixture models. Technical Report 9815, Dept. of Statistics, University of Toronto, 1998.

C.P. Robert and G. Casella. *Monte Carlo Statistical Methods*. New York: Springer-Verlag, second edition, 2004.

J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4, 1994.

S. Thrun and J. O'Sullivan. Discovering structure in multiple learning tasks: The TC algorithm. In *Proceedings of the 13th International Conference on Machine Learning*, 1996.

S. Thrun and L.Y. Pratt, editors. *Learning To Learn*. Kluwer Academic Publishers, Boston, MA, 1998.

M. West, P. Müller, and M.D. Escobar. Hierarchical priors and mixture models, with application in regression and density estimation. *Aspects of Uncertainty: A Tribute to D.V. Lindley*, pages 363–386, 1994.

K. Yu, A. Schwaighofer, V. Tresp, W.-Y. Ma, and H. Zhang. Collaborative ensemble learning: Combining collaborative and content-based information filtering via hierarchical bayes. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence*, 2003.

K. Yu, V. Tresp, and S. Yu. A nonparametric hierarchical Bayesian framework for information filtering. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2004.

K. Yu, A. Schwaighofer, and V. Tresp. Learning Gaussian processes from multiple tasks. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.

J. Zhang, Z. Ghahramani, and Y. Yang. Learning multiple related tasks using latent independent component analysis. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*. MIT Press, Cambridge, MA, 2006.

Journal of Machine Learning Research 8 (2007) 65-102 Submitted 1/06; Revised 9/06; Published 1/07

A Unified Continuous Optimization Framework for Center-Based Clustering Methods

Marc Teboulle TEBOULLE@MATH.TAU.AC.IL
School of Mathematical Sciences
Tel-Aviv University
Tel-Aviv 69978, Israel

Editor: Charles Elkan

Abstract

Center-based partitioning clustering algorithms rely on minimizing an appropriately formulated objective function, and different formulations suggest different possible algorithms. In this paper, we start with the standard nonconvex and nonsmooth formulation of the partitioning clustering problem. We demonstrate that within this elementary formulation, convex analysis tools and optimization theory provide a unifying language and framework to design, analyze and extend hard and soft center-based clustering algorithms, through a generic algorithm which retains the computational simplicity of the popular k-means scheme. We show that several well known and more recent center-based clustering algorithms, which have been derived either heuristically, or/and have emerged from intuitive analogies in physics, statistical techniques and information theoretic perspectives can be recovered as special cases of the proposed analysis and we streamline their relationships.

Keywords: clustering, k-means algorithm, convex analysis, support and asymptotic functions, distance-like functions, Bregman and Csiszar divergences, nonlinear means, nonsmooth optimization, smoothing algorithms, fixed point methods, deterministic annealing, expectation maximization, information theory and entropy methods

1. Introduction

The clustering problem is to partition a given data set into similar subsets or clusters, so that objects/points in a cluster are more similar to each other than to points in another cluster. This is one of the fundamental problem in unsupervised machine learning, and it arises in a wide scope of applications such as astrophysics, medicine, information retrieval, and data mining to name just a few. A closely related problem is the one of vector quantization, mainly developed in the field of communication/information theory. The interdisciplinary nature of clustering is evident through its vast literature which includes many clustering problem formulations, and even more algorithms. For a survey on clustering approaches, see Jain et al. (1999), and for vector quantization the review paper of Gray and Neuhoff (1998), and the references therein.

Basically, the two main approaches to clustering are hierarchical clustering and partitioning clustering. In this paper we focus on partitioning clustering, where the number of clusters is known in advance. Most well known partitioning clustering methods iteratively update the so-called centroids or cluster centers, and as such, are often referred as *center-based* clustering methods. These clustering methods are usually classified as: *hard* and *soft*, depending on the way data points are assigned to clusters. Hard clustering produces a disjoint partition of the data, that is, a binary strategy is used so that each data point belongs exactly to one of the partitions. In that class, one of the most celebrated and widely used hard clustering algorithm is the classical k-means algorithm, which basic ingredients can already be traced back to an earlier work of Steinhaus (1956). The algorithm has been derived in the statistical literature by Forgy (1965) and MacQueen (1967). Another well known variant includes the Linear Vector Quantization (LVQ) algorithm of Lloyd (1982). Soft clustering is a *relaxation* of the binary strategy used in hard clustering, and allows for overlapping of the partitions. In that class, there exists a plethora of algorithms which will be shortly outlined below.

1.1 Motivation

Finding a true optimal partition of a fixed number of sets in a given *n*-dimensional space is known to be an NP-hard problem (Garey and Johnson, 1979), and thus is in general out of reach. As a result, the current literature abounds in approximation algorithms for the partitioning clustering problem.

This paper is a theoretical study of center-based clustering methods from a continuous optimization theory viewpoint. One of the most basic and well-known formulation of the partitioning clustering problem consists of minimizing the sum of a finite collection of "min"-functions, which is a nonsmooth and nonconvex optimization problem. Building on convex and nonlinear analysis techniques, we present a generic way to design, analyze and extend center-based clustering algorithms by replacing the nonsmooth clustering problem with a smooth optimization problem. The general framework and formalism has the advantage to provide a rigorous analysis for center-based clustering algorithms, as well as to reveal the underlying difficulties, and it paves the way for building new schemes. Moreover, as we shall see below, this provides a closure and unification to many disparate approaches that have led to center-based algorithms which have been widely used in applications, for example, fuzzy k-means, deterministic annealing, clustering with general divergences, and which are shown to be special cases of the proposed framework.

We give now a brief summary of some of the relevant works that have motivated the present study. All current center-based algorithms seek to minimize a particular objective function with an attempt to improve upon the standard k-means algorithm. The latter is very attractive due to its computational simplicity. It begins with an initial guess of the centers (usually at random), and consists of two main steps: the first is the cluster assignment, which assigns each data point to the closest cluster center; the second step re-compute the cluster centers as a weighted arithmetic mean of all points assigned to each cluster centers. The algorithm stops when there is no more changes in the partitions, that is, cluster centers no longer move. The simplicity of the k-means has also a price, and it is well known that besides the fact that it does not find an optimal partition, a fact which is surely not too surprising given the alluded difficulty of the partitioning problem, the algorithm also suffers from several drawbacks, for example, it is highly sensitive to initial choice of cluster centers, it might produce empty clusters, it does not provide flexibility to model and measure the influence of specific data types arising in different applications etc...All these difficulties have motivated the search for "better quality" clustering algorithms that could possibly cope with the listed drawbacks.

To achieve this goal, various approaches and techniques have been advocated and a large body of literature has emerged, in particular in the statistic, computer science and engineering research related disciplines, while in the continuous optimization community, clustering analysis has received limited attention, with some of the early studies including for example Rao (1971) and Gordon and Henderson (1977). One direction of research in optimization has been to consider heuristic extensions of the k-means algorithm for the classical minimum sum of squares clustering problem, and which have been shown to be quite successful in practice. For instance, the work of Hansen and Mladenovic (2001) suggests a new descent local search heuristic and reports experimental results showing that it outperforms other known local search methods, quite substantially. A more recent account of optimization approaches to clustering can be found in the excellent and extensive survey paper of Bagirov et al. (2003), and references therein, which among various nonsmooth and global optimization methods, includes the application of the discrete gradient method. The latter is a technique for local optimization that can escape from stationary points which are not local minimizers (see also for instance the more recent work Bagirov and Ugon, 2005).

Another direction of research has been to consider objective functions involving proximity measures other than the usual squared Euclidean distance that is used in k-means. Indeed, different data types arising in many applications have justified the search for more meaningful proximity measures that can better model a given data set. In hard clustering algorithms, several researchers have considered such an approach to extend the k-means algorithm. To mention just a few of the recent studies in that direction (see, for example, Modha and Spanger, 2003; Banerjee et al., 2005; Teboulle et al., 2006), and the extensive relevant bibliography given in these papers.

More intensive research activities have focused on developing soft clustering algorithms. In that context, the literature on iterative methods for clustering is wide, and includes a large number of works and approaches that have been motivated by different fields of applications, and often use different tools and terminology. Many soft clustering algorithms have emerged and have been developed from heuristic considerations, axiomatic approaches, or/and are based on statistical techniques, physical analogies, and information theoretic perspectives. For example, well known soft clustering methods include the Fuzzy k-means (FKM), (see, for example, Bezdek, 1981), the Expectation Maximization algorithm (EM), (see, for example, Duda et al., 2001), Maximum Entropy Clustering Algorithms (MECA), (see, for example, Rose, 1998), the Deterministic Annealing (DA) (Rose et al., 1990), and the closely related similar technique with the same name proposed by Ueda and Nakano (1998). The latter technique is very useful in practice, see for instance its application to documents clustering in the recent work of Elkan (2006). More recent and other soft clustering methods include for example the work of Zhang et al. (1999) which proposes a clustering algorithm called *k*-harmonic means, and which relies on optimizing an objective function defined as the harmonic mean of the squared Euclidean distance from each data points to all centers. An extension of this method has also been further developed in Hamerly and Elkan (2002). All the cited studies have also reported many experimental results to demonstrate the potential benefits of modifying, and extending the aforementioned classes of center-based algorithms, and to show their promise, and/or advantage over the standard k-means, as well as their relevance in several practical and real-life application contexts.

1.2 Main Contributions

Motivated by all these works, this paper has three main goals: (a) to reveal the underlying mathematical tools that explain and enables us to design, analyze and extend center-based clustering algorithms, (b) to develop a generic iterative scheme that keeps the simplicity of the k-means, allows for a rigorous analysis of center-based clustering methods, and reveals their potential advantages and limitations; (c) to provide a closure and unification to a long list of disparate motivations and approaches that have been proposed for center-based clustering methods, and which as alluded above, have been widely used in practice.

To achieve these goals, in this paper we develop a systematic and theoretical study of center-based clustering methods from a continuous optimization theory perspective, which leads to a common language, and a unifying framework for building and analyzing a broad class of hard and soft center-based clustering algorithms. This provides the basis for significantly extend the scope of center-based partitioning clustering algorithms, to bridge the gap between previous works that were relying on heuristics and experimental methods, and to bring new insights into their potential. The proposed framework also shows that all current center-based algorithms are capable of handling only the nonsmoothness difficulty inherent in the clustering problem, but do not provide a cure to the nonconvexity difficulty which remains a challenging one.

A brief summary of our results and the organization of the paper is a follows. In Section 2 we begin with the standard optimization formulation of the partitioning clustering problem that focuses on the nonsmooth nonconvex optimization formulation in a finite dimensional Euclidean space. An obvious key factor in modelling a clustering problem is the choice of the distance measure involved, and which depends on the nature of the problem's data. To handle this situation, we will consider a broad concept of distance-like functions (Auslender and Teboulle, 2006), which extends (and includes) the usual quadratic Euclidean distance setting. We outline their basic properties, and give two generic examples which include the useful and important Bregman and Csiszar based divergences. In Section 3, we furnish the necessary background and known results from convex analysis, with a particular focus on two central mathematical objects: *support and asymptotic functions*, which will play a primary role in the forthcoming analysis of clustering problems. The connection between support and asymptotic functions has been used in past optimization studies to develop a general approach to smoothing nonsmooth optimization problems (Ben-Tal and Teboulle, 1989; Auslender, 1999; Auslender and Teboulle, 2003). Building on these ideas, in Section 4 we first describe an *exact smoothing* mechanism, which provides a very simple way to design and analyze a wide class of center-based hard clustering algorithms. In turns, the support function formulation of the clustering problem also provides the starting point for developing a new and general *approximate smoothing* approach to clustering problems. This is achieved by combining the notion of asymptotic functions with another fundamental mathematical object: the concept of *nonlinear means* of Hardy et al. (1934). We study the relationships and properties of both concepts, and demonstrate that their combination provides a natural and useful framework in the context of clustering. This enables us to arrive at a unified approach for the formulation and rigorous analysis of soft center-based clustering methods. Building on these results, and thanks to the specific form of the objective function one has to minimize in the resulting smooth reformulation of the clustering problem, in Section 5 we introduce a simple generic fixed point algorithm, analyze its properties and establish its convergence to a stationary point. The generic algorithm is computationally as simple as the k-means method, and thus appears suitable for practical purposes. Finally, in Section 6, we show that all the aforementioned hard and soft center-based clustering methods, which have been proposed in the literature from different motivations and approaches can be realized as special cases of the proposed analysis, and we streamline their relationships.

Notations. We use boldface notation for a finite collection of k vectors in a given finite dimensional Euclidean space $\mathbb{R}^n$, that is, $\mathbb{R}^{kn} \ni \mathbf{x} := (x^1, \ldots, x^k)$, with $\mathbb{R}^n \ni x^l := (x^l_1, \ldots x^l_n)$, $l = 1, \ldots, k$. The inner product for two vectors u, v in $\mathbb{R}^n$ is denoted by $u^T v \equiv \langle u, v \rangle$. For an open set $S \subset \mathbb{R}^n$, the notation $\overline{S}$ stands for the topological closure of S, and we also use the notation $\mathbf{S}$, $(\overline{\mathbf{S}})$ to denote the k-

fold Cartesian product $S \times \ldots \times S$, $(\bar{S} \times \ldots \times \bar{S})$. For any nonempty convex set $C \subset \mathbb{R}^n$, δ_C denotes the indicator function of C, $\operatorname{int} C$ ($\operatorname{ri} S$) its interior (relative interior). The convex hull of a set $\mathcal{A}$ is denoted by $\operatorname{conv} \mathcal{A}$. The set of vectors in $\mathbb{R}^n$ with nonnegative (positive, negative) components is denoted by $\mathbb{R}^n_+$ ($\mathbb{R}^n_{++}, \mathbb{R}^n_{--}$). For any function g defined on $\mathbb{R}^n$, we also use the notation $g(z) \equiv g(z_1, \ldots, z_n)$.

2. The Clustering Problem with General Distance-Like Functions

In this paper, we focus on the basic nonsmooth nonconvex optimization formulation of the partitioning clustering problem which uses a broad class of distance-like functions (proximity measures) that replaces (and includes) the usual squared Euclidean norm. As we shall see later on in Sections 4 and 5, this formulation provides a source of explanations to design and analyze center-based clustering iterative algorithms.

2.1 Nonsmooth Optimization Formulation of Clustering

Let $\mathcal{A} = \{a^1, \ldots, a^m\}$ be a given set of points in the subset S of a finite dimensional Euclidean space $\mathbb{R}^n$, and let $1 < k < m$ be a fixed given number of clusters. The clustering problem consists of partitioning the data $\mathcal{A}$ into k subsets $\{A_1, \ldots, A_k\}$, called clusters. The common approach to formulate the clustering problem is as follows. For each $l = 1, \ldots, k$, the cluster A_l is represented by its center (centroid) x^l, and we want to determine k cluster centers $\{x^1, \ldots x^k\}$ such that the sum of proximity measures from each point a^i to a nearest cluster center x^l is minimized. Suppose for the moment that we are given a proximity measure $d(\cdot, \cdot)$ that satisfies the following basic properties:

$$d(u,v) \geq 0, \ \forall (u,v) \in S \ \text{ and } \ d(u,v) = 0 \iff u = v.$$

We will call $d(\cdot, \cdot)$ a *distance-like* function, since we are not necessarily asking for $d(\cdot, \cdot)$ to be symmetric or to satisfy the triangle inequality, (a more precise definition for d is given later in § 2.2). Then, the distance from each $a^i \in \mathcal{A}$ to the closest cluster center is:

$$D(\mathbf{x}, a^i) = \min_{1 \leq l \leq k} d(x^l, a^i).$$

The clustering problem seeks to minimize the average over the entire data set $\mathcal{A}$. Thus, assigning a positive weight ν_i to each $D(\mathbf{x}, a^i)$, such that $\sum_{i=1}^m \nu_i = 1$, (for example, each ν_i can be used to model the relative importance of each point a^i), the clustering problem consists of finding the set of k centers $\{x^1, \ldots, x^k\}$ that solves

$$(NS) \quad \min_{x^1, \ldots, x^k \in S} F(x^1, \ldots, x^k) := \sum_{i=1}^m \nu_i \min_{1 \leq l \leq k} d(x^l, a^i).$$

When the distance is the square of the Euclidean norm, that is, $d(u,v) = \|u - v\|^2$, and the average is the special uniform case, that is, $\nu_i = m^{-1}$ for all i, this formulation can be traced back to the work of Lloyd (1982) when related to vector quantization algorithms (Linde et al., 1980).

For $k = 1$, problem (NS) can either be analytically solved (e.g., when d is the quadratic norm) or is just an easy convex problem, when the proximity measure d is given convex. Likewise, for $k = m$, the (NS) problem is trivial, (i.e., each point is assigned to each cluster), while for $k > m$ the problem is infeasible. Thus, the interesting situation is when $1 < k < m$, for which the problem (NS) is nonsmooth and nonconvex. Furthermore, the number of variables is $n \times k$, and since n is typically very large, even with a moderate number of clusters k, the clustering problem yields a very large scale optimization problem to be solved. Therefore, the clustering problem in this most elementary formulation (i.e., where k, the number of clusters is known) combines three of the most difficult and challenging characteristics one encounters in an optimization problem: *nonsmooth, nonconvex and large scale*.

2.2 Clustering with General Distance-Like Functions

We introduce a broad class of distance-like functions d that is used to formulate the clustering problem in its general form, and we provide two generic families of such distance-like functions for clustering. To measure the proximity of two given vectors in some subset S of $\mathbb{R}^n$, we consider the following concept of distance-like functions, as defined in the recent work of Auslender and Teboulle (2006). The later concept has been widely used in several optimization algorithms, and for more details and results we refer the reader to that work and the references therein.

First, we need to recall some basic notations and definitions in convex analysis, (see, for example, Rockafellar, 1970). For a convex function $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, its effective domain is defined by $\operatorname{dom} g = \{u \mid g(u) < +\infty\}$, and the function is called proper if $\operatorname{dom} g \neq \emptyset$, and $g(u) > -\infty$, $\forall u \in \operatorname{dom} g$. For a proper, convex and lower semicontinuous function (lsc) g, (that is to say, that the epigraph of g is a closed set in $\mathbb{R}^n \times \mathbb{R}$) its subdifferential at x is defined by $\partial g(x) = \{\gamma \in \mathbb{R}^n \mid g(z) \geq g(x) + \langle \gamma, z - x\rangle, \ \forall z \in \mathbb{R}^n\}$ and we set $\operatorname{dom} \partial g = \{x \in \mathbb{R}^n \mid \partial g(x) \neq \emptyset\}$. When g is differentiable, the subdifferential set reduces to the singleton $\nabla g(x)$, the gradient of g at x. Equipped with these notations, we define now the notion of distance-like function.

Definition 1 *A function $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+ \cup \{+\infty\}$ is called a distance-like function with respect to an open nonempty convex set $S \subset \mathbb{R}^n$ if for each $y \in S$ it satisfies the following properties:*
(d_1) *$d(\cdot, y)$ is proper, lsc, convex, and C^2 on S with a positive definite symmetric matrix Hessian*[1] *denoted by $\nabla^2 d(\cdot, y)$;*
(d_2) *$\operatorname{dom} d(\cdot, y) \subset \overline{S}$, and $\operatorname{dom} \partial_1 d(\cdot, y) = S$ where $\partial_1 d(\cdot, y)$ denotes the subgradient map of the function $d(\cdot, y)$ with respect to the first variable;*
(d_3) *$d(\cdot, y)$ is level bounded on $\mathbb{R}^n$, that is, $\lim_{\|u\| \to \infty} d(u, y) = +\infty$;*
(d_4) *$d(y, y) = 0$, (which also implies that $\nabla_1 d(y, y) = 0$, where $\nabla_1 d(\cdot, y)$, is the gradient with respect to the first variable.)*

We denote by $\mathcal{D}(S)$ the family of functions d satisfying the premises of Definition 1. It should be emphasized that the triangle inequality, and symmetry is not required in the definition of d, hence the use of the "distance-like" terminology.

To understand the motivation behind the technical assumptions of Definition 1, and the forthcoming mathematical developments given in Sections 4 and 5, let us already announce as an appetizer, that the main (essentially the only one) computational step that will be needed in the generic algorithm we developed in this paper for solving the clustering problem (NS), reduces to the solution of an optimization problem which admits the simple form:

$$\min\{\sum_{i=1}^{m} \gamma_i d(x, a^i) \mid x \in \overline{S}\}, \tag{1}$$

1. We recall that the positive definiteness of the Hessian matrix implies that $d(\cdot, y)$ is strictly convex.

with some given $\gamma_i > 0,\ i = 1, \dots, m$. The properties (d_1), and (d_3) are needed to guarantee the existence of a unique global minimizer x^* to the optimization problem of the form (1), while property (d_2) plays the role of a "barrier", and enforces the optimal solution x^* to be in the open set S, (see Section 5.1, and Lemma 14 for details). Property (d_4) is just for normalization of d.

The class $\mathcal{D}(S)$ has been chosen in our setup, since it is broad and includes two useful generic families that produces a wide variety of distance-like functions which are discussed next. However, it should be noticed that as long as we can guarantee the existence of a unique global minimizer to problem (1), the use of other classes of distance-like functions is possible; this situation will be further illustrated below in Example 3.

2.3 Generic Families of Distance-Like Functions

As special cases of the class $\mathcal{D}(S)$, we briefly recall two particularly useful generic families that produce a wide variety of distance-like functions which have been already shown to be relevant for clustering, see for instance the recent work of Teboulle et al. (2006), and references therein.

• **Bregman Based Distances Type**[2] Originally proposed by Bregman (1967), this class of distances have been widely extended and analyzed in several optimization contexts and methods by Censor and his co-authors, (see, for example, Censor and Lent, 1981), and the more recent comprehensive monograph of Censor and Zenios (1997).

Let $\psi : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ be a proper, lsc, strictly convex function, with $\operatorname{dom}\psi \subset \overline{S}$, and such that ψ is continuously differentiable on $S := \operatorname{int}(\operatorname{dom}\psi)$. The Bregman based distance associated with ψ is the function $d_\psi : \mathbb{R}^n \times \mathbb{R}^n \to [0, +\infty]$ defined by

$$d_\psi(x,y) := \begin{cases} \psi(x) - \psi(y) - \langle x - y, \nabla\psi(y)\rangle & \text{if } y \in S \\ +\infty & \text{otherwise.} \end{cases}$$

Most interesting and useful Bregman based distances functions are separable, and can be generated with the choice

$$\psi(x) = \sum_{j=1}^{n} \omega(x_j), \tag{2}$$

with ω being some appropriate *scalar* twice differentiable convex function with $\omega''(\cdot) > 0$ on $\operatorname{int}(\operatorname{dom}\omega)$.

Example 1 Typical choices for $\omega(\cdot)$ include $\omega(t) = t^2, t\log t, -\log t, -(1-t^2)^{1/2}$ with domain $\operatorname{dom}\omega = \mathbb{R}, \mathbb{R}_+, \mathbb{R}_{++}, [-1,1]$, respectively. Substituting these functions in d_ψ, with ψ as defined in (2), the first three choices yields respectively the squared Euclidean distance, the Kullback-Liebler based relative entropy and the Burg based relative entropy (also called the Itakura Saito distance). For more examples and details in the context of clustering, see Teboulle et al. (2006), and references therein.

• **φ-Divergence Based Distances Type**. Originally introduced by Csiszar (1967) in the context of information theory to provide a notion generalizing divergence between probability measures, (e.g., the Kullback-Liebler relative entropy). It has been considered for optimization and algorithmic purposes by Teboulle (1987, 1992, 1997).

2. Another terminology is Bregman divergence. In this paper we will freely use/exchange both terminologies.

Let $\varphi : \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ be a proper, lsc, convex function such that $\operatorname{dom} \varphi \subset \mathbb{R}_+, \operatorname{dom} \partial\varphi = \mathbb{R}_{++}$, and such that φ is C^2, strictly convex, and nonnegative on $\mathbb{R}_{++}$, with $\varphi(1) = \varphi'(1) = 0$. Then, the φ-divergence based distance is defined by,

$$d_\varphi(x, y) := \sum_{j=1}^{n} y_j \varphi\left(\frac{x_j}{y_j}\right),$$

and which by its definition is already separable.

Example 2 Typical examples for the φ-divergence, d_φ with $S = \mathbb{R}^n_{++}$, include

$$\begin{aligned} \varphi_1(t) &= -\log t + t - 1, \text{ (Arithmetic Mean)}, \\ \varphi_2(t) &= t \log t - t + 1, \text{ (Geometric Mean)}, \\ \varphi_3(t) &= 2(\sqrt{t} - 1)^2, \text{ (Root Mean Square)}. \end{aligned}$$

The names in parenthesis indicate the type of *means* that results by solving problem of the form (1) in these cases, (Teboulle et al., 2006).

For the above choices, and other appropriate ψ or/and φ, and S, both functionals d_ψ, d_φ can be shown to be in the class $\mathcal{D}(S)$ as defined in Definition 1.[3] In particular, it can be easily seen that $d \equiv d_\psi$ or d_φ enjoy the required basic property of a distance-like function, namely one can verify that:

$$\forall (u, v) \in S \times S \;\; d(u, v) \geq 0 \text{ and } d(u, v) = 0 \text{ iff } u = v.$$

Moreover, note that the distance-like function d_φ is jointly convex on $S \times S$, and hence in particular, for any $u \in S$, the function $v \to d_\varphi(u, v)$ is convex in its *second argument*, a property which is not shared in general by the Bregman-based distance.

As previously noticed, symmetry is not requested in the definition of d, and as a result, as long as we can guarantee the existence of a global minimizer for problem of the form (1), some of the properties requested in Definition 1 with respect to the first argument, can be exchanged with respect to the second argument, or even relaxed. This situation is further exemplified now.

Example 3 In a very recent paper, Banerjee et al. (2005) have proposed to use Bregman distance for clustering by considering it as function with respect to its second argument (i.e., by changing the order of the variables), and have derived an interesting and somewhat surprising result. More precisely, consider problem (1) with

$$d(x, a) := d_\psi(a, x) := \psi(a) - \psi(x) - \langle a - x, \nabla\psi(x) \rangle, \;\; \forall (a, x) \in \bar{S} \times S,$$

where $S = \operatorname{int}(\operatorname{dom} \psi)$. In that case, in general, $x \to d_\psi(a, x)$ is not necessarily convex,[4] and hence the corresponding problem (1) is *nonconvex*. Even though problem (1) is nonconvex, it has been recently shown (Banerjee et al., 2005), that if a minimizer of (1) exists in S, then it is a *global*

3. Note that when no confusion occurs, and with some abuse of notations, throughout the paper, d_φ (d_ψ) always stands for the φ-divergence (Bregman divergence).

4. Exceptions include the well known special cases, when the Bregman divergence is the squared Euclidean distance and the relative entropy. Otherwise, the convexity of $x \to d_\psi(a, x)$ is not warranted without imposing further conditions on ψ, which unfortunately precludes the use of relevant or/and useful ψ for defining a Bregman divergence.

minimizer, and is always the (weighted) *arithmetic mean* of the data. This can be seen as follows. A very simple, but fundamental property satisfied by any Bregman distance, is the following three point identity revealed in Chen and Teboulle (1993), (see Lemma 3.1 there), and which naturally generalizes Pythagoras theorem. For any three points $u, v \in \operatorname{int}(\operatorname{dom}\psi)$ and $w \in \operatorname{dom}\psi$ the following identity holds true

$$d_\psi(w,v) - d_\psi(w,u) = d_\psi(u,v) + \langle \nabla\psi(u) - \nabla\psi(v), w - u\rangle. \tag{3}$$

Thanks to the identity (3), and the fact that $d_\psi(\cdot,\cdot) \geq 0$, one has for any $i = 1,\dots,m$:

$$\begin{aligned} d_\psi(a^i,x) - d_\psi(a^i,z) &= d_\psi(z,x) + \langle \nabla\psi(z) - \nabla\psi(x), a^i - z\rangle, \\ &\geq \langle \nabla\psi(z) - \nabla\psi(x), a^i - z\rangle. \end{aligned}$$

Thus, multiplying the last inequality by $\gamma_i > 0$, and summing over $i = 1,\dots,m$, it immediately follows that with $z := \sum_{i=1}^n \gamma_i a^i$, the right hand side of the inequality vanishes, and hence

$$\sum_{i=1}^m \gamma_i d_\psi(a^i,x) \geq \sum_{i=1}^m \gamma_i d_\psi(a^i,z), \ \forall x \in S,$$

showing that $z \in S$ is the global minimizer of problem (1).

Note that it is also possible to consider other classes of distance-like functions, which are not necessarily based on Bregman or/and φ divergences (as long as a unique global optimal solution of (1) is warranted). An interesting recent study can be found for example, in the work of Modha and Spanger (2003) which have considered convex-k-means clustering algorithms based on some other proximity measures that are convex in the second argument.

Finally, note also that one could easily enrich the model (NS) by considering for example a more general formulation that associates with each l a different distance $d_l \in \mathcal{D}(S_l)$ (and which can be useful in applications to accommodate different types of data, Modha and Spanger, 2003), so that the more general model would consist of solving

$$\min\left\{ \sum_{i=1}^m \nu_i \min_{1\leq l\leq k} d_l(x^l,a^i) \mid (x^1,\dots,x^k) \in S_1 \times \cdots \times S_k \right\}.$$

The analysis and theoretical results that we developed below also hold for such more general formulations as well.

3. Convex Analysis Background: Support and Asymptotic Functions

The main approach in this paper is based on considering ways to replace the nonsmooth clustering problem (NS) via a *smooth* optimization problem, and to study and derive a corresponding generic algorithm solving the nonsmooth problem (NS) via its smoothed counterpart. To develop this approach, this section furnishes some of the key concepts and results of convex analysis that will be used throughout this paper. For further details and proofs of the material presented in this section (see for example Rockafellar 1970, Sections 12 and 13, and Auslender and Teboulle 2003, Chapter 2). Readers familiar with these concepts may skip directly to Section 4, perhaps after reading the examples of the present section which provide motivation to the forthcoming developments.

3.1 Support Functions

A fundamental concept for dealing with properties of closed convex sets is the concept of support function. It allows to transfer properties about sets via functions, and as a result turns out to play a central role in optimization problems and facilitate their analysis.

Definition 2 *For any set $C \subset \mathbb{R}^k$, the function $\sigma_C : \mathbb{R}^k \to [-\infty, +\infty]$ defined by*

$$\sigma_C(v) := \sup\{\langle u, v\rangle \mid u \in C\}, \tag{4}$$

is called the support function of C.

Geometrically, the support function of a set C describes the closed half-spaces which contain C, namely

$$C \subset \{u \mid \langle u, v\rangle \le \alpha\} \iff \sigma_C(v) \le \alpha.$$

Note that the supremum in (4) may be finite or infinite; attained on C or not. If $C = \emptyset$, we set $\sigma_C \equiv -\infty$, while if $C \neq \emptyset$, one has $\sigma_C > -\infty$ and $\sigma_C(0) = 0$.

For the forthcoming analysis, we consider the case when C is a closed convex set in $\mathbb{R}^k$. The support function can be computed for many interesting geometric convex sets C, (Rockafellar, 1970, Section 13, p. 113). Let us give here two particularly interesting examples.

Example 4 Let $C := \mathbf{B} = \{u \in \mathbb{R}^k \mid \|u\| \le 1\}$ be the unit Euclidean ball. Then, applying Cauchy-Schwartz inequality, it is easy to verify that

$$\sigma_{\mathbf{B}}(v) = \|v\|,$$

that is, the Euclidean norm is the support function of the unit ball.

Example 5 Let C be the unit simplex in $\mathbb{R}^k$, that is,

$$C := \Delta = \{u \in \mathbb{R}^k \mid \sum_{j=1}^{k} u_j = 1, u_j \ge 0,\ j = 1, \dots, k\}.$$

Then, a simple computation shows that

$$\sigma_\Delta(v) = \sup\{\langle u, v\rangle \mid u \in \Delta\} = \max_{1\le j\le k} v_j,$$

the supremum being attained on the compact set Δ, at $\{u_l^* : l = 1, \dots, k\}$ given by:

$$u_l^* = \begin{cases} 1 & \text{if } l = \operatorname*{argmax}_{1\le j\le k} v_j \\ 0 & \text{otherwise.} \end{cases}$$

At this stage, and to motivate the reader for the forthcoming technical results and analysis, let us already emphasize that support functions are essentially "built-in" for most optimization problems. More precisely, most smooth and nonsmooth optimization problems can be modelled via the following generic abstract optimization model

$$\inf\{c_0(u) + \sigma_Y(c(u)) \mid c(u) \in \operatorname{dom} \sigma_Y\},$$

where $c(u) = (c_1(u), \dots, c_m(u)) \in \mathbb{R}^m$, where all $\{c_i\}_{i=0}^m$ are real valued on $\mathbb{R}^k$, and $Y \subset \mathbb{R}^m$ is adequately defined. The re-formulation of optimization problems via their support functions provides an alternative way to view and tackle optimization problems by exploiting mathematical properties of support functions, and this is the line of analysis that will be used here for the clustering problem. The following examples illustrate the support function formulations of some generic classes of optimization problems. More details can be found in Auslender and Teboulle (2003, Chapter 2).

Example 6 (a) (Nonlinear programming). Consider the standard optimization problem

$$v_{NLP} := \inf\{c_0(u) \mid c_i(u) \le 0,\ i = 1, \dots, m,\ u \in \mathbb{R}^k\}.$$

Then, it is easy to see that with $Y = \mathbb{R}^m_+$, one has

$$v_{NLP} = \inf_{u \in \mathbb{R}^k} \{c_0(u) + \sup\{\langle y, c(u)\rangle \mid y \in \mathbb{R}^m_+\}\} = \inf_{u \in \mathbb{R}^k} \{c_0(u) + \sigma_{\mathbb{R}^m_+}(c(u))\}.$$

Note that the first equation above is nothing else but the Lagrangian representation of (NLP), with $y \in \mathbb{R}^m_+$ being the Lagrangian multiplier associated with the constraints.
(b) (l_p-norm optimization problems). Consider the problem

$$v_{LP} := \inf\{\|c(u)\|_p,\ u \in \mathbb{R}^k\},\ \ (p \ge 1),$$

where $\|z\|_p := (\sum_{i=1}^m |z_i|^p)^{1/p}$ is the usual l_p-norm of $z \in \mathbb{R}^m$. Let Y be the l_q-unit ball in $\mathbb{R}^m$, that is,

$$Y := \{y \in \mathbb{R}^m\ \|y\|_q \le 1\},\ \text{with}\ p + q = pq.$$

Then, invoking Hölder inequality, it follows that $\sigma_Y(c(u)) = \|c(u)\|_p$, and hence,

$$v_{LP} = \inf\{\sigma_Y(c(u)) \mid u \in \mathbb{R}^k\}.$$

(c) (Finite minimax problems). Consider the finite minimax problem

$$v_{MM} := \inf_{u \in \mathbb{R}^k} \max_{1 \le i \le m} c_i(u).$$

Then, using Example 5 with $Y = \Delta$, the unit simplex in $\mathbb{R}^m$, one obtains

$$v_{MM} := \inf\{\sigma_Y(c(u)) \mid u \in \mathbb{R}^k\}.$$

The above examples have illustrated the primary role of support functions in formulating optimization problems. Our goal will be to exploit the special structure and further properties of support functions for deriving useful equivalent reformulations of the clustering problem. The next result recorded below in Theorem 4 reveals an important and relevant property of support functions.

First, let us recall that the support function of a given set C is intimately connected to the well-known indicator of the set C, through another fundamental operation in convex analysis, which is the conjugacy operation.

Definition 3 *For any function $g : \mathbb{R}^k \to \mathbb{R} \cup \{+\infty\}$, the convex conjugate of g is the function $g^* : \mathbb{R}^k \to \mathbb{R} \cup \{+\infty\}$ defined by*

$$g^*(z) = \sup_{y \in \mathbb{R}^k} \{\langle y, z\rangle - g(y)\} = \sup_{y \in \operatorname{dom} g} \{\langle y, z\rangle - g(y)\}.$$

In addition, if g is proper, lower semicontinuous (lsc) and convex on $\mathbb{R}^k$, then so is its conjugate g^*, and $g^{**} = g$.

Now, consider the indicator function of C defined by

$$\delta_C(u) = \begin{cases} 0 & \text{if } u \in C \\ +\infty & \text{otherwise.} \end{cases}$$

Then, using the definition of the support function, one has

$$\sigma_C(v) = \sup_{u \in C} \langle u, v \rangle = \sup_{u \in \mathbb{R}^k} \{ \langle u, v \rangle - \delta_C(u) \}.$$

Thus, we see that σ_C is nothing else but the conjugate of the indicator function δ_C, that is, $\sigma_C = \delta_C^*$. Moreover, if C is also closed convex, then

$$\delta_C^{**} = \delta_C = \sigma_C^*.$$

Therefore, the indicator function and the support function of a closed convex set are conjugate to each other. In fact, the support function provides a remarkable one-to-one correspondence between nonempty closed convex subsets of $\mathbb{R}^k$ and the class of positively homogeneous lsc proper convex functions through the conjugacy operation.[5] This result is formally cited below.

Theorem 4 *(Rockafellar, 1970, Theorem 13.2). The functions which are the support functions of non-empty convex sets are the lsc proper convex functions which are positively homogeneous.*

3.2 Asymptotic Functions

We now look at the relationship between the support function and another important mathematical object, called the asymptotic function. For that purpose, we first need to define the notion of asymptotic cone.[6]

The asymptotic cone of a nonempty convex set $C \subset \mathbb{R}^k$ is a convex cone containing the origin defined by,

$$C_\infty := \{ v \in \mathbb{R}^k : v + C \subset C \}.$$

Geometrically, this means that the asymptotic convex cone C_∞ includes the origin and consists of all directions $v \in \mathbb{R}^k$, such that for each $u \in C$, the halfline $\{u + tv \mid t \geq 0\}$ is contained in C. This notion is useful for dealing with unbounded sets, namely when we are concerned in specifying directions in which a set is unbounded. For example, one can show that a nonempty closed convex subset C of $\mathbb{R}^k$ is bounded if and only if $C_\infty = \{0\}$. Here, we are interested in the behavior of convex functions "in the large", that is, in the way convex functions vary, as their argument move along halflines in $\mathbb{R}^k$. A convenient way to achieve this is through the notion of asymptotic function which is just the asymptotic cone of the epigraph of that function. Intuitively speaking, the result given below says that the asymptotic behavior of g along halflines depends only on the direction of the halfline, and not on its location.

5. Recall that a function p is positively homogeneous on $\mathbb{R}^k$ if $0 \in \operatorname{dom} p$ and $p(tu) = tp(u)$ for all $u \in \mathbb{R}^k$ and all $t > 0$.
6. In the convex setting (i.e., when working with convex sets and convex functions, the asymptotic cone (function) is often called recession cone (function). In fact for closed convex sets (functions), the two concepts coincide (Auslender and Teboulle, 2003).

Definition 5 *Let $g : \mathbb{R}^k \to \mathbb{R} \cup \{+\infty\}$ be a proper, convex and lower semicontinuous (lsc) function. The asymptotic function g_∞ of g is the function $g_\infty : \mathbb{R}^k \to \mathbb{R}$ defined by*

$$epi\ (g_\infty) = (epi\, g)_\infty,$$

where $epi\, g = \{(x,r) \in \mathbb{R}^k \times \mathbb{R} : g(x) \le r\} \subset \mathbb{R}^{k+1}$ is the epigraph of g.

The next result shows that this definition makes sense, and collect some basic properties of the asymptotic function g_∞.

Proposition 6 *(Rockafellar, 1970, Theorem 8.5) Let $g : \mathbb{R}^k \to \mathbb{R} \cup \{+\infty\}$, be a proper, convex function. Then its asymptotic function g_∞ is a proper convex function on $\mathbb{R}^k$ that is positively homogeneous with $g_\infty(0) = 0$. For any $z \in \mathbb{R}^k$, one has*

$$g_\infty(z) = \sup\{g(u+z) - g(u) \mid u \in \operatorname{dom} g\}.$$

Furthermore, if g is also assumed lsc on $\mathbb{R}^k$, then g_∞ is also lsc, and for any $u \in \operatorname{dom} g$ and any $z \in \mathbb{R}^k$,

$$g_\infty(z) = \sup_{t>0} t^{-1}[g(u+tz) - g(u)] = \lim_{t\to+\infty} t^{-1}[g(u+tz) - g(u)].$$

In particular, one also has

$$g_\infty(z) = \lim_{s\to 0^+} \left\{g_s(z) := sg(s^{-1}z)\right\}, \ \forall\, \mathbb{R}^k \ni z \in \operatorname{dom} g. \tag{5}$$

The last property (5) is useful to compute asymptotic functions. To illustrate this, let us give a few interesting examples (see, for example Example 2.5.1, page 51 in Auslender and Teboulle, 2003).

Example 7 In the following three examples, it can be verified that $g(\cdot)$ is a proper lsc convex function on $\mathbb{R}^k$, and thus we can use (5) to compute $g_\infty(\cdot)$.
(a) Let $g(u) = \sqrt{1 + \|u\|^2}$. Then, using (5) one has $g_\infty(z) = \|z\|$.
(b) Let $g(u) = \sum_{j=1}^k e^{u_j}$. Then one obtains $g_\infty(z) = \delta_{\mathbb{R}^k_-}$, that is, the indicator of the negative orthant in $\mathbb{R}^k$.
(c) Let $g(u) = \log \sum_{j=1}^k e^{u_j}$. Then, $g_\infty(z) = \max_{1\le j\le k} z_j$.

To expand our ability of computing the asymptotic function g_∞ of a given function g, it turns out that it is often easier to work with the conjugate of g, which is always lsc, and therefore, by Proposition 6 so is its asymptotic function. Since this asymptotic function is lsc, proper, positively homogeneous and convex, Theorem 4 guarantees that it must be the support function of some nonempty closed convex set. The next convex analytic result shows that this set should be precisely the effective domain of the conjugate g^*.

Proposition 7 *(Rockafellar, 1970, Theorem 13.3) Let $g : \mathbb{R}^k \to \mathbb{R} \cup \{+\infty\}$ be a proper convex and lsc function, and let $C = \operatorname{dom} g^*$ be the effective domain of the conjugate g^*. Then, $\sigma_C = g_\infty$.*

This last result connecting support and asymptotic functions, together with (5) in Proposition 6, provides the basis and motivation for developing a general approach to smoothing nonsmooth optimization problems. This approach was introduced by Ben-Tal and Teboulle (1989), and for more general results and details, the reader is refereed to Auslender (1999), and the recent monograph of Auslender and Teboulle (2003). Building on these ideas, we now develop smoothing approaches to the clustering problem.

4. Smoothing Methodologies for Clustering

We first describe a very simple *exact* smoothing mechanism which provides a novel and simple way to view and design all center-based *hard* clustering algorithms from an optimization perspective. In turns, the support function formulation also provides the starting point for developing a new and general smoothing approach for clustering problems, which is based on combining asymptotic functions, and the fundamental notion of *nonlinear means* of Hardy et al. (1934). The resulting smoothing approach lends to devise a simple generic algorithm, which computationally is as simple as the popular k-means scheme, (see Section 5), and encompasses and extend most well known *soft* clustering algorithms, see Section 6.

4.1 Exact Smoothing: The Support Function Approach and Hard Clustering

Given $\{a^1,\dots,a^m\}$ in the subset $S \subset \mathbb{R}^n$, and a distance-like function $d \in \mathcal{D}(S)$, the general clustering problem as formulated in Section 2 through (NS) is to solve

$$\min_{x^1,\dots,x^k \in S} F(x^1,\dots,x^k) \equiv \min_{\mathbf{x}\in\mathbf{S}} F(\mathbf{x}) := \sum_{i=1}^m \nu_i \min_{1\le l\le k} d(x^l,a^i),$$

where we use the notation $\mathbf{x} = (x^1,\dots,x^k)$, for the $k \times n$ dimensional vector $\mathbf{x}$, and $\mathbf{S} \subseteq \mathbb{R}^N$, with $N := kn$, for the k-fold Cartesian product of S.

In the context of the clustering problem, we now briefly show that the support function allows to derive an equivalent smooth formulation of the clustering problem, and in fact provides the foundation to the design and analysis of hard clustering algorithms. Fix any $i \in \{1,\dots,m\}$, and let

$$d^i(\mathbf{x}) := (d(x^1,a^i),\dots,d(x^k,a^i)) \in \mathbb{R}^k.$$

The nonsmooth term $\min_{1\le l\le k} d(x^l,a^i)$ can be replaced by a smooth one, by using the support function. Indeed, using Example 5, it follows that for any $i = 1,\dots,m$,

$$\min_{1\le l\le k} d(x^l,a^i) = -\sigma_{\Delta^i}(-d^i(\mathbf{x})) = \min\{\langle w^i, d^i(\mathbf{x})\rangle : \ w^i \in \Delta^i\}, \tag{6}$$

where Δ^i is the unit simplex in $\mathbb{R}^k$ given by

$$\Delta^i = \left\{ w^i \in \mathbb{R}^k \mid \sum_{l=1}^k w^i_l = 1, w^i_l \ge 0,\, l = 1,\dots,k \right\},$$

and where w^i_l is the "membership" variables associated to cluster A_l, which satisfies: $w^i_l = 1$ if the point a^i is closest to cluster A_l, and $w^i_l = 0$ otherwise. Thus, substituting (6) in (NS), it follows that the nonsmooth clustering problem (NS) is equivalent to the exact smooth problem:

$$(ES) \quad \min_{x^1,\dots,x^k\in S}\ \min_{w^1,\dots,w^m \in \mathbb{R}^k} \left\{ \sum_{i=1}^m \nu_i \sum_{l=1}^k w^i_l d(x^l,a^i) \mid w^i \in \Delta^i,\ i = 1,\dots,m \right\}.$$

The smooth formulation (ES) explains precisely the mechanism of all well known, old and more recent, hard center-based hard clustering algorithms. Indeed, applying the Gauss-Seidel (GS) minimization algorithm to problem (ES), (which is also often called alternative minimization or coordinate descent method, (see, for instance, Auslender, 1976; Bertsekas and Tsitsiklis, 1989; Bertsekas, 99), that is, at each iteration, first minimize with respect to w^i with x^l fixed, then minimize with respect to x^l with the membership variable fixed, yields a general hard clustering algorithm with distance-like functions, which for short is denoted (**HCD**). The algorithm **HCD** includes as special cases, not only the popular k-means algorithm, but also many others hard clustering methods mentioned in the introduction. In particular, it includes and extend the Bregman hard clustering algorithm recently derived by Banerjee et al. (2005, Algorithm 1, p. 1715), which was introduced and motivated from a completely different view point, relying on statistical and information theoretic arguments. To make the paper self-contained, the algorithm **HCD** has been discussed in some details in the appendix.

4.2 Approximate Smoothing via Asymptotic Functions and Soft Clustering

The support function approach which has provided an *exact* smoothed reformulation of the nonsmooth problem (NS) and the corresponding generic hard clustering method **HCD**, lends itself to another systematic way to obtain an *approximate smoothed* reformulation of the problem (NS), which in turn will provide the basis for producing a generic *soft* clustering algorithm.

We have seen in §4.1, that the nonsmooth clustering problem (NS) is equivalent to the exact smooth formulation (ES). Using (6), an equivalent representation of the clustering problem (ES) can also be written as

$$(NS) \quad \min_{x^1,\ldots,x^k \in S} \sum_{i=1}^{m} -\nu_i \sigma_{\Delta_i}(-d(x^1,a^i),\ldots,-d(x^k,a^i))$$

where $\Delta^i := \{w^i \in \mathbb{R}^k : e^T w^i = 1, w^i \geq 0\}$, $i = 1,\ldots,m$, and $e \equiv (1,\ldots 1) \in \mathbb{R}^k$.

Fix any $i \in \{1,\ldots,m\}$. Thanks to Proposition 7, we know that the support function of the set Δ^i corresponds to an asymptotic convex function, say $(g^i)_\infty(\cdot)$. From Proposition 6, this asymptotic function can be approximated (cf. (5)) via:

$$g^i_\infty(z) = \lim_{s\to 0^+} \left\{ g^i_s(z) := s g^i(s^{-1}z) \right\}, \; \forall\, \mathbb{R}^k \ni z \in \operatorname{dom} g^i, \; \forall i = 1,\ldots m,$$

where g^i is some given convex function, such that $\operatorname{dom}(g^i)^* = \Delta^i$. This naturally suggests to replace the support function $-\sigma_{\Delta_i}(\cdot)$ in (NS) by an approximate function $g^i_s(\cdot)$, and thus to consider for each $s > 0$, the following *approximate* problem for (NS):

$$(NS)_s \qquad \min_{x^1,\ldots,x^k \in S} \sum_{i=1}^{m} -\nu_i g^i_s(-d(x^1,a^i),\ldots,-d(x^k,a^i)). \tag{7}$$

Thus, with a function g^i_s smooth enough, this approach leads to a generic smoothed approximate reformulation of the nonsmooth problem (NS) which depends on the parameter $s > 0$ that plays the role of a *smoothing* parameter.

A key question is then to find appropriate candidates for the function g^i for the clustering problem. Answer to this question will be developed in the next two subsections. But first, let us illustrate the above approach with an important example.

Example 8 *(The Log-Sum Exponential Smoothing for Clustering)* The Log-Sum exponential function is an important and very well known operation which has been widely used in optimization contexts, (Bertsekas, 1982; Ben-Tal and Teboulle, 1989). For the clustering problem, it will lead to a family of important methods.

Consider a slight variant, and more general form of the function considered in Example 7(c) given by

$$g(y) = \log \sum_{l=1}^{k} \pi_l e^{y_l}, \tag{8}$$

where π_l are some given weights, that is, $\pi_l \geq 0,\ \forall l = 1,\dots,k$ and $\sum_{l=1}^{k} \pi_l = 1$. This function is convex on $\mathbb{R}^k$, and from Example 7(c) one obtains [7]

$$g_\infty(y) = \lim_{s \to 0} g_s(y) = \max_{1 \leq l \leq k} y_l.$$

Thus, using (8), the clustering problem (NS) can be approximated by solving the smoothed problem $(NS)_s$ which in this case reads:

$$(NS)_s \qquad \min_{x^1,\dots,x^k \in S} F_s(\mathbf{x}) := -s \sum_{i=1}^{m} \nu_i \log \left(\sum_{l=1}^{k} \pi_l e^{-\frac{d(x^l,a^i)}{s}} \right).$$

When d is the squared Euclidean distance, (with $\nu_i = m^{-1}, \forall i,\ \pi_l = k^{-1}, \forall l$), the objective function just derived from the proposed smoothing optimization approach, is in fact exactly the objective function arising in some well known clustering methods, such as the so-called (EM)-algorithm for normal mixtures, (Duda et al., 2001), and the deterministic annealing (Rose et al., 1990), which were motivated by, and derived from, a statistical/probabilistic framework and statistical physics analogies. Further, when d is a Bregman function, as given in Example 3, the approximation model $(NS)_s$ yields precisely the Bregman soft clustering method recently derived by Banerjee et al. (2005) from an information theory view point. This will be further discussed in Section 6.

As we shall see next, another natural way to smooth the clustering problem, and which later on will reconcile with the asymptotic function approach, is by using the so-called concept of nonlinear means.

4.3 Approximate Smoothing via Nonlinear Means

The concept of nonlinear means defined below, was introduced in 1934 by Hardy et al. (1934, Chapter III) as a natural generalization of the well known power means, that is, the weighted l_p-norm of a positive vector z.

Definition 8 *Let $h : \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ be a strictly increasing and convex function, and let h^{-1} be its inverse, which is thus strictly increasing and concave. The nonlinear mean of k real numbers $z_1,\dots,z_k$ associated to h is defined by*

$$G_h(z) = h^{-1}\left(\sum_{l=1}^{k} \pi_l h(z_l) \right),$$

where π_l are weights which are arbitrary positive numbers whose sum is one, that is, $\pi \in \mathrm{ri}(\Delta)$.

7. In fact, one also has $\operatorname{dom} g^* = \Delta$, (see Lemma 18), and thus as promised by Proposition 7, $\sigma_\Delta = g_\infty$.

In the rest of this paper we use the notation Δ_+ for the relative interior of the simplex in $\mathbb{R}^k$, that is, $\Delta_+ := \{\pi \in \mathbb{R}^k \mid \sum_{l=1}^{k} \pi_l = 1, \pi > 0\}$. As in Hardy et al. (1934, Chapter III, p. 66) we adopt the convention that $h^{-1}(\infty) = b$, where $b := \sup\{t \mid h(t) < \infty\}$.

More details and many results on nonlinear means can be found in Hardy et al. (1934, Chapter III). At this juncture, it is interesting to note that Karayiannis (1999) has suggested an *axiomatic approach* to re-formulate clustering objective functions that essentially leads him to rediscover the notion of nonlinear means given in Hardy et al. (1934). This interesting axiomatic approach can thus be further viewed as an additional supportive argument to the general smoothing optimization approach we develop here.

The next simple result shows that the nonlinear mean does provide an approximation, more precisely a lower bound for the maximum function $\max\limits_{1\le l\le k} z_l$.

Lemma 9 *For each $z \in \mathbb{R}^k$ and any $\pi \in \Delta_+$ the following inequalities hold,*

$$\sum_{l=1}^{k} \pi_l z_l \le G_h(z) \le \max_{1\le l\le k} z_l. \tag{9}$$

Proof. By the convexity of the function h one has

$$h\left(\sum_{l=1}^{k} \pi_l z_l\right) \le \sum_{l=1}^{k} \pi_l h(z_l),$$

and since h^{-1} is increasing, this proves the left hand side inequality of (9). To prove the right hand side of the inequality, note that since $\sum\limits_{l=1}^{k} \pi_l z_l \le \max\limits_{1\le l\le k} z_l$, and h is increasing, then

$$\sum_{l=1}^{k} \pi_l h(z_l) \le \max_{1\le l\le k} h(z_l) = h\left(\max_{1\le l\le k} z_l\right),$$

and an application of h^{-1} to both sides of the inequality completes the proof. ■

Recall that our basic clustering problem (NS) consists of minimizing the objective function F which can be rewritten as:

$$F(\mathbf{x}) = \sum_{i=1}^{m} \nu_i \min_{1\le l\le k} d(x^l, a^i) = -\sum_{i=1}^{m} \nu_i \max_{1\le l\le k} \{-d(x^l, a^i)\}. \tag{10}$$

Since $d(\cdot,\cdot) \ge 0$, to approximate $\max_l -d(x^l, a^i)$ we need only to consider nonlinear means with domain containing the negative orthant $\mathbb{R}^k_-$.

Example 9 *(Nonlinear Means on $\mathbb{R}^k_-$.)* Consider the functions $h_i(t), i = 1,2,3$ given respectively by $-\log(-t), -t^{-1}, -\sqrt{-t}$ with domain $(-\infty, 0)$ for the first two, and $(-\infty, 0]$ for the last one. The corresponding nonlinear means G_{h_i} are then respectively the weighted geometric, harmonic, and square root mean of $z \in \mathbb{R}^k_{--}$, for the first two choices of h, and of $z \in \mathbb{R}^k_-$ for the last one, while if $z \notin \mathbb{R}^k_{--}$ ($z \notin \mathbb{R}^k_-$ for the last one), one has $G_h(z) = 0$.

In view of the upper bound of Lemma 9, for each i, we can consider approximating the quantity $\max_{1\le l\le k}(-d(x^l,a^i))$, by its nonlinear mean G_h, and hence the resulting objective F given in (10) by an approximate objective given by:

$$\hat{F}(\mathbf{x}) = -\sum_{i=1}^{m} \nu_i h^{-1}\left(\sum_{l=1}^{k} \pi_l h(-d(x^l,a^i))\right). \tag{11}$$

Let us illustrate this on two specific examples with some h as given in Example 9.

Example 10 *(Harmonic Mean Approximation)* Consider the function $h_2(t) = -t^{-1}$ from Example 9, with $\operatorname{dom} h_2 = \operatorname{int}\operatorname{dom} h_2 = (-\infty, 0)$, that yields the harmonic mean G_{h_2}. Then, using h_2 in (11), to approximate F given in (10), one has to consider minimizing the approximate objective:

$$\hat{F}(\mathbf{x}) = \sum_{i=1}^{m} \nu_i \left(\sum_{i=1}^{k} \frac{\pi_l}{d(x^l,a^i)}\right)^{-1}.$$

This recovers and extends the approximate objective $\hat{F}(\mathbf{x})$, (with $d(\cdot,\cdot)$ the squared Euclidean distance, $\pi_l = k^{-1}$, $\forall l$, $\nu_i = m^{-1}$, $\forall i$), which was recently suggested by Zhang et al. (1999) from heuristic and intuitive considerations, together with a corresponding *k-harmonic means* algorithm, and some interesting numerical results. More recently, Hamerly and Elkan (2002) have further studied new variants of the k-harmonic means algorithm, and have experimentally shown its superiority for finding clustering of high quality in low dimensions. However, no mathematical or/and convergence analysis of the proposed algorithms have been provided in these works. It will be shown later on, that the k-harmonic means algorithm can also be viewed as a particular realization of our generic algorithm for which our convergence result can be applied, see Section 5.

Example 11 *(Geometric Mean Approximation)* Take the function $h_1(t) = -\log(-t)$ given in Example 9 with $\operatorname{dom} h_1 = \operatorname{int}\operatorname{dom} h_1 = (-\infty, 0)$ that yields the geometric means G_{h_1}. Then, using (11), we then obtain as an approximation of F, the resulting approximate objective:

$$\hat{F}(\mathbf{x}) = \sum_{i=1}^{m} \nu_i \prod_{l=1}^{k} d(x^l,a^i)^{\pi_l}.$$

This example provides an apparently new approximate model for clustering, on which one can apply the generic scheme developed in Section 5.

Now, we return to the Log-Sum exponential function described in Example 8. With the choice $h(t) = e^t$, in Definition 8, the resulting nonlinear means G_h precisely recovers the convex Log-Sum exponential function,

$$G_h(z) = \log \sum_{l=1}^{k} \pi_l e^{z_l}.$$

Therefore, since in that case $G_h(\cdot)$ is convex, Proposition 6 can be applied, and one has:

$$\max_{1\le l\le k} z_l = \lim_{s\to 0} s G_h(s^{-1}z) = G_h^{\infty}(z),$$

where the later expression denotes the asymptotic function of the mean $G_h(\cdot)$.

Thus, this specific example shows that the objective function of the clustering problem (NS) (cf. (10)), can be approximated either by a nonlinear (smooth) mean, or, by the corresponding asymptotic function of $G_h(\cdot)$, provided the later can be well defined. This suggests an approach that would combine nonlinear means and asymptotic functions to provide a generic smoothing model which will approximate the original clustering problem (NS), and approach it in the limit, for a broad class of smooth approximations. This can be achieved, provided we can characterize the class of functions h for which $G_h(\cdot)$ remains convex. This is developed next.

4.4 Combining Asymptotic Functions and Convex Nonlinear Means

The mean $G_h(\cdot)$ being by definition the composition of a convex function with a concave one, is not necessarily convex. Thus, defining its corresponding asymptotic function as given in Proposition 6 is not warranted. Furthermore, it turns out that the convexity of $G_h(\cdot)$ plays a crucial role in the convergence proof of the forthcoming generic algorithm (see Section 5). Thus, it is important to characterize the convexity of the nonlinear mean $G_h(\cdot)$. By specializing a general result proven in Ben-Tal and Teboulle (1986, Theorem 2.1), we can identify a wide class of functions h for which $G_h(\cdot)$ is convex.

In the sequel, for convenience we will often use the notation $\Omega := \text{int}(\text{dom}\, h)$, and Ω^k to denote the k-fold Cartesian product of $\text{int}(\text{dom}\, h)$.

Lemma 10 *Let $h : \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ be C^3 on Ω, strictly increasing and convex, and let $r(t) := -h''(t)/h'(t)$. Then $G_h(z)$ is convex on Ω^k if and only if $1/r(t)$ is convex on Ω.*

Proof. Define $\xi(z) := \sum_{l=1}^{k} \pi_l h(z_l)$, so that $G_h(z) = h^{-1}(\xi(z))$. Then, G_h is convex if and only if it satisfies the gradient inequality, that is, recalling that $(h^{-1})'(\cdot) > 0$, this is equivalent to say,

$$\frac{h^{-1}(\xi(y)) - h^{-1}(\xi(x))}{(h^{-1})'(\xi(x))} \geq \sum_{l=1}^{k} \pi_l (y_l - x_l) h'(x_l), \ \ \forall x, y \in \Omega^k. \tag{12}$$

To prove that (12) holds true, define,

$$H(s,t) := \frac{h^{-1}(s) - h^{-1}(t)}{(h^{-1})'(t)}.$$

Invoking Ben-Tal and Teboulle (1986, Lemma 1, p. 1449), one has H is concave in (s,t) if and only if $1/r(t)$ is convex. To complete the proof it thus remains to show that the concavity of H reduces to the validity of (12). Applying Jensen's Inequality for the concave function $H(\cdot,\cdot)$ at $s := \xi(y), t := \xi(x)$, it follows that the function H is concave if and only if,

$$H(\xi(y), \xi(x)) \geq \sum_{l=1}^{k} \pi_l H(h(y_l), h(x_l)) = \sum_{l=1}^{k} \pi_l \frac{(y_l - x_l)}{(h^{-1})'(h(x_l))}.$$

Therefore, using the relation $h'\left(h^{-1}(t)\right) = 1/(h^{-1})'(t)$ at $t := h(x_l)$, proves that the last inequality is exactly (12). ■

Define the class of functions,

$$\mathcal{H} := \left\{ h \in C^3(\Omega) \mid h' > 0, \ h'' > 0, \text{ and } \frac{1}{r} \text{ is convex} \right\}.$$

The class of functions $\mathcal{H}$ satisfying the condition $1/r(t)$ is convex is wide, (Ben-Tal and Teboulle, 1986). It includes in particular, all functions h such that $1/r(t)$ is a linear function, that is, $1/r(t) = at+b,\ a,b \in \mathbb{R}$.

Example 12 *(Convex Nonlinear Means)* It can be easily verified that the class of functions $h \in \mathcal{H}$ satisfying $1/r(t) = at+b$ for some $a,b \in \mathbb{R}$ includes in particular (with $a=0,\ b \neq 0$) the function $h(t) = e^{t/b}$, as well as, for example, the functions h with $\Omega \equiv \operatorname{int}(\operatorname{dom} h) = (-\infty,\delta),\ (\delta \geq 0))$ given by
(i) $h(t) = -(\delta - t)^p,\ p \in (0,1),\ (a = (p-1)^{-1},\ b = \delta(1-p)^{-1})$.
(ii) $h(t) = (\delta - t)^{-p},\ p \in (0,+\infty),\ (a = -(p+1)^{-1},\ b = \delta(p+1)^{-1})$.
(iii) $h(t) = -\log(\delta - t), (a = -1,\ b = \delta)$.
Clearly these examples for h include and extend all the previous choices $h_i(t),\ i = 1,2,3$ (cf. Example 9), and generate accordingly corresponding *convex* nonlinear means G_h.

Equipped with Lemma 10, the concept of *asymptotic nonlinear mean* associated to a given convex nonlinear mean G_h is now well defined through Proposition 6.

Definition 11 *Let $h \in \mathcal{H}$. For $z \in \mathbb{R}^k$ and $\pi \in \Delta_+$, the asymptotic nonlinear mean is defined by*

$$G_h^\infty(z) = \lim_{s \to 0^+} s h^{-1}\left(\sum_{l=1}^k \pi_l h\left(\frac{z_l}{s}\right)\right).$$

With a proof identical to that of Lemma 9 we immediately get the following result.

Lemma 12 *If $h \in \mathcal{H}$, then for each $z \in \mathbb{R}^k$ and any $\pi \in \Delta_+$ one has*

$$\sum_{l=1}^k \pi_l z_l \leq G_h^\infty(z) \leq \max_{1 \leq l \leq k} z_l.$$

This last result, together with Definition 11, and the results developed in Section 4.2 (cf. (7)), provide all the ingredients to combine nonlinear convex means as characterized in Lemma 10 with asymptotic functions, and to formulate a broad class of smooth approximations to the clustering problem (NS) as follows.

For any $h \in \mathcal{H}$, any fixed $s > 0$, and any $\pi \in \Delta_+$, we approximate the nonsmooth objective F of the original clustering problem (NS) by the smooth function:

$$F_s(x^1,\ldots,x^k) \equiv F_s(\mathbf{x}) = -s\sum_{i=1}^m \nu_i h^{-1}\left(\sum_{l=1}^k \pi_l h\left(\frac{-d(x^l,a^i)}{s}\right)\right).$$

In the rest of the paper, we focus on developing and analyzing a generic algorithm that minimizes the smoothed approximate nonconvex function $F_s(\cdot)$.

Remark 13 Lemma 9 and Lemma 12 show that the nonlinear mean and its asymptotic version always provides a lower bound for $\max_{1 \leq i \leq k} z_i$, and hence when applied to the function F_s it follows that for any $s > 0$,

$$\sum_{i=1}^m \nu_i \left\{\min_{1 \leq l \leq k} d(x^l,a^i) - \sum_{l=1}^k \pi_l d(x^l,a^i)\right\} \leq F(\mathbf{x}) - F_s(\mathbf{x}) \leq 0. \tag{13}$$

The quality of the smooth approximation is somewhat hidden in the last inequality. Yet, it is important to note that the Log-Sum exponential mean generated via $h(t) = e^t$ appears to be a sort of *optimal mean* for approximating the finite $\max_{1 \le l \le k} z_l$ function. Indeed, take for example $\nu_i = m^{-1}$, $\forall i$, and $\pi_l = k^{-1}$, $\forall k$. In that case,

$$F_s(\mathbf{x}) = s \log k - \frac{s}{m} \sum_{i=1}^{m} s \log \left(\sum_{l=1}^{k} e^{\frac{-d(x^l, a^i)}{s}} \right) := s \log k + L_s(\mathbf{x}).$$

Then, the right inequality (13) produces the well known approximation result for $L_s(\cdot)$ (Bertsekas, 1982; Ben-Tal and Teboulle, 1989):

$$0 \le F(\mathbf{x}) - L_s(\mathbf{x}) \le sm \log k,$$

showing that the Log-Sum exponential mean shares a unique type of uniform approximation, and for which $G_h^\infty(z) \equiv \max_{1 \le i \le k} z_i$. More on this specific property of the Log-Sum exponential function will be discussed in Section 6.

5. The Smooth k-Means Algorithm: Properties and Convergence

Building on the previously developed results, in this section we present a simple generic center-based algorithm for soft clustering, that we call the Smooth k-means (**SKM**) algorithm, and we study its convergence properties.

5.1 Motivation

Given $d \in \mathcal{D}(S)$, $h \in \mathcal{H}$ and any $s > 0$, to solve the clustering problem we consider a solution method that solves the approximate smoothed minimization problem,

$$\inf \left\{ F_s(\mathbf{x}) \mid \mathbf{x} \in \overline{\mathbf{S}} \right\}, \tag{14}$$

where

$$F_s(\mathbf{x}) = -s \sum_{i=1}^{m} \nu_i h^{-1} \left(\sum_{l=1}^{k} \pi_l h \left(-\frac{d(x^l, a^i)}{s} \right) \right). \tag{15}$$

This problem could be solved by some standard optimization algorithms, such as projected gradient/Newton type methods, Lagrangian multipliers, etc. (see, for example, Bertsekas, 99). However, given that clustering problems are usually very large scale, we are interested to devise a simple iterative scheme which does not require any sophisticated computations at each iteration, (e.g., Hessian computations, matrix inversions, or/and line search techniques), which are usually needed in the alluded standard optimization algorithms.

It turns out that the specific form of F_s lends itself to build a simple iterative scheme, by combining the smoothing approach with successive approximations. The idea of such combination is well known in the field of optimization, and can be traced back to the so-called Weiszfeld algorithm derived in 1937 for solving some basic location theory problems (Weiszfeld, 1937). The Weiszfeld algorithm has provided a fertile ground for many other algorithms and problems in a variety of research areas, (see, for example, Ben-Tal et al., 1991; Brimberg and Love, 1993, and many of the references cited therein).

To motivate the generic algorithm SKM described below in §5.2, let us consider for the moment the special case when $\mathbf{S} = \mathbb{R}^N$, with the distance like function between any two points $u, v \in \mathbb{R}^n$ being the usual squared Euclidean distance $d(u,v) = \|u-v\|^2$.

The necessary local optimality condition for solving problem (14) in that case consists of finding $\mathbf{x} \in \mathbb{R}^N$ satisfying

$$\nabla F_s(\mathbf{x}) = 0. \tag{16}$$

Recall that we use the notation $\mathbf{x} = (x^1, \dots, x^k)$ with $x^l \in \mathbb{R}^n$, $l = 1, \dots, k$, and $N = kn$. We denote by $\nabla_l F_s(\mathbf{x})$ the gradient of $F_s(\mathbf{x})$ with respect to $x^l \in \mathbb{R}^n$. To express (16) in a compact and informative way we also use the following notations. For any scalar $s > 0$, $i = 1, \dots, m$ and $l = 1, \dots, k$, let

$$\delta^i(x^l) \quad := \quad -s^{-1} d(x^l, a^i), \text{ with } \delta^i(\mathbf{x}) = (\delta^i(x^1), \dots \delta^i(x^k)) \in \Omega^k, \tag{17}$$

$$\rho^{il}(\mathbf{x}) \quad := \quad \pi_l \frac{h'(\delta^i(x^l))}{h'(G_h(\delta^i(\mathbf{x})))}. \tag{18}$$

With (17), since $h'(\cdot) > 0$ and $\pi_l > 0, \forall l$, the functions $\rho^{il}(\cdot)$ are positive for every i, l. Now, using the definition of F_s given in (15), we obtain for each $l = 1, \dots, k$

$$\nabla_l F_s(\mathbf{x}) = \pi_l \sum_{i=1}^m \nu_i \left(h^{-1}\right)' \left(\sum_{l=1}^k \pi_l h \left(-\frac{d(x^l, a^i)}{s} \right) \right) \cdot h' \left(-\frac{d(x^l, a^i)}{s} \right) \nabla_1 d(x^l, a^i), \tag{19}$$

where $\nabla_1 d(x^l, a^i)$ is the gradient of d with respect to the first variable x^l. Using in (19) the relation

$$(h^{-1})'(t) = \frac{1}{h'\left(h^{-1}(t)\right)},$$

the definition of $G_h(\cdot)$, and (18), simple algebra shows that (16) reduces to

$$\nabla_l F_s(\mathbf{x}) = \sum_{i=1}^m \nu_i \rho^{il}(\mathbf{x}) \nabla_1 d(x^l, a^i) = 0, \ l = 1, \dots, k. \tag{20}$$

Since for the moment, we assumed $d(u,v) = \|u-v\|^2$, then $\nabla_1 d(u,v) = 2(u-v)$, and (20) simplifies to

$$\sum_{i=1}^m \nu_i \rho^{il}(\mathbf{x})(x^l - a^i) = 0, \ l = 1, \dots, k. \tag{21}$$

Defining for each $i = 1, \dots, m$, and $l = 1, \dots, k$,

$$\lambda^{il}(\mathbf{x}) := \nu_i \rho^{il}(\mathbf{x}) \cdot \left(\sum_{j=1}^m \nu_j \rho^{jl}(\mathbf{x}) \right)^{-1}, \tag{22}$$

one has $\lambda^{il}(\mathbf{x}) > 0$, and $\sum_{i=1}^m \lambda^{il}(\mathbf{x}) = 1$, and (21) reduces to

$$x^l = \sum_{i=1}^m \lambda^{il}(\mathbf{x}) a^i, \qquad l = 1, \dots, k. \tag{23}$$

Formula (23) suggests that in order to find x^l, we can consider the following fixed point iteration: for $t = 0, 1, \dots,$

$$x^l(t+1) = \sum_{i=1}^{m} \lambda^{il}(\mathbf{x}(t))\, a^i, \quad l = 1, \dots, k. \tag{24}$$

The explicit formula for $\mathbf{x}(t+1)$ given in (24) has been achieved thanks to our ability to solve the equation (20) for $\mathbf{x}$, which in turns, follows from the *linearity* of the gradient map $\nabla_1 d(\cdot,\cdot)$, for the special case of the squared Euclidean distance. In fact, the fixed point iteration (24) obtained from solving (20) reads equivalently as:

$$x^l(t+1) = \operatorname*{argmin}_{x^l} \left\{ \sum_{i=1}^{m} \nu_i \rho^{il}(\mathbf{x}(t)) d(x^l, a^i) \right\}, \; l = 1, \dots, k.$$

This provides the motivation for the extension given next, and which allows to handle the general case with distance-like functions.

5.2 The SKM Algorithm

Mimicking the approach just outlined in the special case of the squared Euclidean distance, this naturally suggests that for handling general distance-like function $d \in \mathcal{D}(S)$, one computes $\mathbf{x}(t+1) = (x^1(t+1), \dots, x^k(t+1))$ by solving:

$$\mathbf{x}(t+1) = \operatorname*{argmin}_{\mathbf{x}} \left\{ \sum_{i=1}^{m} \sum_{l=1}^{k} \nu_i \rho^{il}(\mathbf{x}(t)) d(x^l, a^i) \right\} \equiv \operatorname*{argmin}_{\mathbf{x} \in \overline{\mathbf{S}}} A_s(\mathbf{x}, \mathbf{x}(t)),$$

where $A_s : \mathbb{R}^N \times \mathbf{S} \to \mathbb{R}_+ \cup \{+\infty\}$ is defined by

$$A_s(\mathbf{x}, \mathbf{u}) = \sum_{i=1}^{m} \sum_{l=1}^{k} \nu_i \rho^{il}(\mathbf{u}) d(x^l, a^i), \tag{25}$$

and with $\{\rho^{il}(\cdot)\}_{i,l} > 0$ as defined in (18)) for some given $h \in \mathcal{H}$, (A_s, depends on $s > 0$ through the definition of ρ^{il}).

This leads us to propose the following simple generic family of iterative algorithms for solving (14).

The SKM Algorithm. For given data points $\{a^1, \dots, a^m\} \in S \subset \mathbb{R}^n$, pick a distance like function $d \in \mathcal{D}(S)$, a function $h \in \mathcal{H}$ and fix $s > 0$. Set $t = 0$, choose $\mathbf{x}(0) \in \mathbf{S}$ and generate iteratively the sequence $\{\mathbf{x}(t)\}_{t=0}^{\infty}$ by solving:

$$\mathbf{x}(t+1) = \operatorname{argmin} \left\{ A_s(\mathbf{x}, \mathbf{x}(t)) \mid \mathbf{x} \in \overline{\mathbf{S}} \right\},$$

until convergence.

The next result shows that the algorithm **SKM** is well defined.

Lemma 14 *Let $d \in \mathcal{D}(S)$, $h \in \mathcal{H}$ and $s > 0$. For any fixed $\mathbf{u} \in \mathbf{S}$, consider the convex optimization problem*

$$(P_u) \quad v(\mathbf{u}) := \inf\{A_s(\mathbf{x}, \mathbf{u}) \mid \mathbf{x} \in \overline{\mathbf{S}}\},$$

where $A_s(\mathbf{x},\mathbf{u})$ is defined in (25). Then, the following statements hold:
(i) The optimal set $X^*(\mathbf{u})$ of problem (P_u) is nonempty and compact.
(ii) There exists a unique minimizer $\mathbf{y}(\mathbf{u}) := (y^1(\mathbf{u}),\dots,y^k(\mathbf{u})) \in \mathbf{S}$ solving (P_u) and satisfying

$$\sum_{i=1}^{m} \nu_i \rho^{il}(\mathbf{u}) \nabla_1 d(y^l(\mathbf{u}), a^i) = 0, \;\; l = 1,\dots,k. \tag{26}$$

Proof. (i) For any fixed $\mathbf{u} \in \mathbf{S}$, and $s > 0$, let $\Phi_u(\mathbf{x}) := A_s(\mathbf{x},\mathbf{u}) + \sum_{l=1}^{k} \delta\left(x^l | \overline{S}\right)$, where $\delta(\cdot|\overline{S})$ denotes the indicator of $\overline{S}$. Then, by the definition of $A_s(\cdot,\mathbf{u})$ given in (25), and property (d2) in Definition 1, the minimization problem (P_u) can be written as

$$v(\mathbf{u}) := \inf\{\Phi_u(\mathbf{x}) : \; \mathbf{x} \in \mathbb{R}^N\}.$$

Since here $v(\mathbf{u})$ is finite, recalling the definition of the distance-like function d (cf. Definition 1), it follows by (d_3) that $\Phi_u(\cdot)$ is level bounded. Therefore with $\Phi_u(\cdot)$ being a proper, lsc, and convex function, it follows that the optimal set $X^*(\mathbf{u})$ of problem (P_u) is nonempty and compact, and hence the existence of a minimizer is warranted.
(ii) The minimizer is unique thanks to the strict convexity of $A_s(\cdot,\mathbf{u})$ (which is implied from (d_1) of Definition 1, recalling that $\nu_i > 0$ and $\rho^{il}(\cdot) > 0$ for every i,l). From the optimality conditions, for each $y(\mathbf{u}) \in X^*(\mathbf{u})$ we have $0 \in \partial\Phi_u(y(\mathbf{u}))$, where $\partial\Phi_u$ stands for the subdifferential of Φ_u. Then, applying Rockafellar (1970, Theorem 23.8), it follows that for each $l = 1,\dots,k$ the optimality of $\mathbf{y}(\mathbf{u})$ yields

$$0 \in \sum_{i=1}^{m} \nu_i \partial_1 d(y^l(\mathbf{u}), a^i) \rho^{il}(\mathbf{u}) + N_{\overline{S}}(y^l(\mathbf{u})), \tag{27}$$

where $N_{\overline{S}}(y^l(\mathbf{u}))$ stands for the normal cone[8] to $\overline{S}$ at $y^l(\mathbf{u})$. Since by definition, $\nu_i > 0$, $\rho^{il}(\mathbf{u}) > 0$, $\forall i,l$, and since by (d_2) of Definition 1, for each $i \in [1,m]$, one has $\operatorname{dom} \partial_1 d(\cdot,a^i) = S$ a nonempty open convex set, it follows that $\mathbf{y}(\mathbf{u}) \in \mathbf{S}$, and $N_{\overline{S}}(y^l(\mathbf{u})) = \{0\}$, and hence (27) reduces to the desired equation (26). ■

The main computational step of the algorithm **SKM** consists of solving for $x^l(t+1)$ the equation

$$\sum_{i=1}^{m} \nu_i \rho^{il}(\mathbf{x}(t)) \nabla_1 d(x^l(t+1), a^i) = 0, \qquad l = 1,\dots,k. \tag{28}$$

As already mentioned (cf. §2.2), the class of distance-like functions which are separable includes most interesting and useful examples based on Bregman divergences, while φ-divergences are by definition given separable. More generally, let us consider now what will be called the class of *separable* distance-like functions, with d defined by

$$d(x,y) = \sum_{j=1}^{n} \theta(x_j, y_j), \tag{29}$$

8. For a closed convex set $C \subset \mathbb{R}^n$, recall Rockafellar (1970) that the normal cone to C at $x \in C$ is defined by $N_C(x) = \partial\delta_C(x) = \{\nu \in \mathbb{R}^n \mid \langle \nu, z - x\rangle \le 0, \forall z \in C\}$.

where $\theta : \mathbb{R} \times \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ satisfies the premises of Definition 1 (i.e., with $S := I$, and I being an open interval). For such a separable d the equation (28) reduces to solving for each $l = l, \ldots k$, the simple *scalar* equation in the variable $x^l_j(\cdot)$:

$$\sum_{i=1}^{m} \nu_i \rho^{il}(\mathbf{x}(t))\theta'(x^l_j(t+1), a^i_j) = 0,\ j = 1, \ldots, n, \tag{30}$$

where $\theta'(\cdot, a^i_j)$ is the derivative with respect to the first argument. Both separable Bregman divergences, and φ-divergences are then recovered as special cases of (29) with

$$\theta(\alpha, \beta) := \psi(\alpha) - \psi(\beta) - (\alpha - \beta)\psi'(\beta) \text{ and } \theta(\alpha, \beta) := \beta\varphi(\frac{\alpha}{\beta}),$$

respectively, and with the appropriate scalar functions ψ, φ (cf. §2.2). Note that the equation (30) can be solved *analytically* for $x^l(t+1)$ in the case of Bregman divergences, as well as for the case of φ-divergences for many interesting choices of φ, see, for example, Example 2, and for more examples and details, the recent work (Teboulle et al., 2006), and references therein.

Thanks to the strict monotonicity of $\theta'(\cdot, a^i_j)$ in its first variable (inherited from Definition 1), we can establish for the class of separable distance-like functions defined in (29), the following property of **SKM**, which simple proof is left to the reader.

Proposition 15 *Let $d \in \mathcal{D}(S)$ be separable and let $\{\mathbf{x}(t)\}_{t=0}^{\infty}$ be the sequence generated by* **SKM**. *Then, for each $l = 1, \ldots, k$, the iterates $S \ni x^l(t+1)$, $t = 0, 1, \ldots$ that solves (30), satisfy:*

$$\min_{1 \le i \le m} a^i_j \le x^l_j(t+1) \le \max_{1 \le i \le m} a^i_j,\ j = 1, \ldots, n,$$

that is, the sequence $\{\mathbf{x}(t)\}_{t=0}^{\infty}$ lies in a bounded hypercube in $\mathbf{S}$.

We end this section with two additional remarks on the computational aspects of **SKM**.
(a) The computational complexity of **SKM** is as simple as the standard k-means algorithm, which makes **SKM** viable for large scale clustering problems, and allows to significantly extend the scope of soft center-based iterative clustering methods.
(b) Although this paper is concerned with theoretical issues, it should also be noted that in a more practical implementation of **SKM**, one could also start at $t = 0$ with a fixed positive value for s_t and decrease iteratively the parameter s_t. For that purpose, various strategies from standard optimization techniques, such as for example within penalty/barrier methods can be considered, (see, for example, Bertsekas, 1982, 99). Yet, recall that any fixed $s > 0$ do provide an approximate solution as well, as explained in §4.3, see, for example, Example 10.

5.3 Convergence Analysis

We are now in the position to state and prove the main convergence result for **SKM**. Note that the key element in the proof strongly relies on the convexity result established in Lemma 10 for the nonlinear means $G_h(\cdot)$.

Theorem 16 *Let $\{\mathbf{x}(t)\}_{t=0}^{\infty}$ be the sequence generated by the* **SKM** *algorithm. Then,*
(i) $F_s(\mathbf{x}(t+1)) < F_s(\mathbf{x}(t))$, for all $\mathbf{x}(t+1) \neq \mathbf{x}(t)$.
(ii) Let $d \in \mathcal{D}(S)$ be separable. Then, the sequence $\{\mathbf{x}(t)\}_{t=0}^{\infty}$ is bounded and each limit point $\mathbf{x} \in \mathbf{S}$ of this sequence is a stationary point for F_s.

Proof (i) By definition of the algorithm **SKM** and Lemma 14, $\mathbf{x}(t+1) \in \mathbf{S}$ is the unique minimizer solving $\mathbf{x}(t+1) = \operatorname*{argmin}_{\mathbf{x}} A_s(\mathbf{x}, \mathbf{x}(t))$, and hence

$$A_s(\mathbf{x}(t+1), \mathbf{x}(t)) < A_s(\mathbf{x}(t), \mathbf{x}(t)), \ \forall\, \mathbf{x}(t+1) \neq \mathbf{x}(t). \tag{31}$$

Let

$$\begin{aligned} v_t^{il} &:= -s^{-1} d(x^l(t), a^i) \in \Omega, \ i = 1, \dots, m, \ l = 1, \dots, k, \\ \text{and } v_t^i &:= (v_t^{i1}, \dots, v_t^{ik}) \in \Omega^k, \ i = 1, \dots, m. \end{aligned} \tag{32}$$

Then,

$$F_s(\mathbf{x}(t)) = -s \sum_{i=1}^m \nu_i h^{-1} \left(\sum_{l=1}^k \pi_l h(v_t^{il}) \right) = -s \sum_{i=1}^m \nu_i G_h(v_t^i). \tag{33}$$

Since $h \in \mathcal{H}$, $s > 0$ and $\nu_i > 0$, then by Lemma 10, $G_h(\cdot)$ is convex on Ω^k. Therefore, applying the gradient inequality to the convex function $G_h(\cdot)$ one has:

$$G_h(z) - G_h(y) \geq \langle z - y, \nabla G_h(y) \rangle, \ \forall y, z \in \Omega^k. \tag{34}$$

Using the points given by $z := v_{t+1}^i$, and $y := v_t^i$, and noting that the l-th component of the gradient of $G_h(\cdot)$ is given by

$$\left(\nabla G_h(v_t^i)\right)_l = \pi_l \frac{h'(v_t^{il})}{h'\left(G_h(v_t^i)\right)} = \rho^{il}(\mathbf{x}(t)), \ l = 1, \dots, k,$$

(where the last equality follows from (18)), one obtains after substituting these expressions in (34),

$$G_h(v_t^i) - G_h(v_{t+1}^i) \leq \sum_{l=1}^k \nu_i \left(v_t^{il} - v_{t+1}^{il} \right) \rho^{il}(\mathbf{x}(t)), \ i = 1, \dots, m.$$

Multiplying by $s > 0$ the above inequality, summing over $i = 1, \dots, m$, using (33) and (32) it follows that for all $\mathbf{x}(t+1) \neq \mathbf{x}(t)$,

$$\begin{aligned} F_s(\mathbf{x}(t+1)) - F_s(\mathbf{x}(t)) &\leq s \sum_{i=1}^m \sum_{l=1}^k \nu_i (v_t^{il} - v_{t+1}^{il}) \rho^{il}(\mathbf{x}(t)), \\ &= \sum_{i=1}^m \sum_{l=1}^k \nu_i d(x^l(t+1), a^i) \rho^{il}(\mathbf{x}(t)) - \sum_{i=1}^m \sum_{l=1}^k \nu_i d(x^l(t), a^i) \rho^{il}(\mathbf{x}(t)), \\ &= A_s(\mathbf{x}(t+1), \mathbf{x}(t)) - A_s(\mathbf{x}(t), \mathbf{x}(t)) < 0, \end{aligned}$$

where the last inequality is from (31), and (i) is proved.
(ii) Let $d \in \mathcal{D}(S)$ be separable. Then, by Proposition 15, the sequence $\{\mathbf{x}(t)\}_{t=0}^\infty$ is bounded with limit points in $\mathbf{S}$. Thus, there exists $\bar{\mathbf{x}} \in \mathbf{S}$ and a convergent subsequence $\{\mathbf{x}(t_j)\}$ to $\bar{\mathbf{x}}$, namely $\lim_{j\to\infty} \mathbf{x}(t_j) = \bar{\mathbf{x}}$ with $\lim_{j\to\infty} t_j = +\infty$. Let $\mathbf{x}^* \in \mathbf{S}$ be a stationary point of F_s, then one has,

$$\sum_{i=1}^m \nu_i \nabla_1 d((x^l)^*, a^i) \rho^{il}(\mathbf{x}^*)) = 0, \ l = 1, \dots, k. \tag{35}$$

We need to show that $\bar{\mathbf{x}} = \mathbf{x}^*$. Thanks to Lemma 14(ii), and by definition of algorithm **SKM**, given $\mathbf{x}(t) \in \mathbf{S}$, there exists a unique $\mathbf{x}(t+1) \in \mathbf{S}$ which satisfies

$$\sum_{i=1}^{m} \nu_i \nabla_1 d(x^l(t+1), a^i) \rho^{il}(\mathbf{x}(t)) = 0, \; l = 1, \ldots, k. \tag{36}$$

Let us denote the solution of (36) by $x^l(t+1) := T^l(\mathbf{x}(t)), \; l = 1, \ldots, k$, and for any $\mathbf{u} \in \mathbf{S}$, set $T(\mathbf{u}) := \left(T^1(\mathbf{u}), \ldots, T^k(\mathbf{u})\right)$. Since $\mathcal{D}(S) \ni d(\cdot, a_i)$ is given C^2 on S, with positive definite matrix Hessian $\nabla_1^2 d(\cdot, a^i)$, and since $\nu_i \rho^{il}(\mathbf{x}(t)) > 0$, then $\sum_{i=1}^{m} \nu_i \nabla_1^2 d(\cdot, a^i) \rho^{il}(\mathbf{x}(t))$ is also a positive definite matrix. Therefore, by invoking the implicit function theorem, it follows that T is continuous on $\mathbf{S}$. Now, by definition of **SKM**, using (36) and (35), in terms of T, it follows that:

$$\mathbf{x}(t) = \mathbf{x}^* \text{ if and only if } T(\mathbf{x}(t)) = \mathbf{x}(t). \tag{37}$$

To complete the proof, we need to consider two cases. First, if for some $t \geq 0, \mathbf{x}(t+1) = T(\mathbf{x}(t)) = \mathbf{x}(t)$, then in that case the sequence repeats itself from that point, and one has $\bar{\mathbf{x}} = \mathbf{x}(t)$, which by (37) implies that $\bar{\mathbf{x}} = \mathbf{x}^*$. In the other case, with $\mathbf{x}(t+1) \neq \mathbf{x}(t)$, by (i) one has for all $t \geq 0, F_s(\mathbf{x}(t+1) < F_s(\mathbf{x}(t))$. Therefore, the sequence $\{F_s(\mathbf{x}(t)\}$ is monotone decreasing, and since $F_s(\mathbf{x}) \geq F(\mathbf{x}) \geq 0$, being also bounded below, it must converge to some limit, and it follows that,

$$\lim_{j \to \infty} [F_s(\mathbf{x}(t_j)) - F_s(T(\mathbf{x}(t_j)))] = 0. \tag{38}$$

Since T is continuous, one also have $\lim_{j \to \infty} T(\mathbf{x}(t_j)) = T(\bar{\mathbf{x}})$, and hence together with (38) we obtain $F_s(\bar{\mathbf{x}}) = F_s(T(\bar{\mathbf{x}}))$. Therefore, by (i), the last equation implies that $\bar{\mathbf{x}} = T(\bar{\mathbf{x}})$, and hence by (37), $\bar{\mathbf{x}} = \mathbf{x}^*$. ■

Remark 17 It should be noted (as already explained in §2.3), that as long as a unique global minimizer of $\mathbf{x} \to A(\mathbf{x}, \mathbf{u})$ exists, and the continuity of the map $T(\cdot)$ on $\mathbf{S}$ can be ensured, a close inspection of the proof reveals that the convergence result established in Theorem 16 could also be used for other classes of distance-like functions, and in particular for the important class of Bregman divergences considered by Banerjee et al. (2005), and discussed in Example 3.

6. Relations with Known Center-Based Clustering Algorithms and Extensions

The purpose of this section is not an intent to survey all the current existing approaches and center-based clustering methods. Rather, our aim is to briefly demonstrate that many of the seemingly different methods cited in the introduction, and which have emerged from various view points, can be derived, analyzed and extended under the unified smoothing optimization approach we have developed in this paper. This is now illustrated below, with a particular focus on the Deterministic Annealing algorithm (DA) and its possible extensions.

Before proceeding, we recall our setting. As outlined in Section 2, there exists essentially two equivalent ways to formulate the clustering problem: the nonsmooth formulation and its equivalent exact smooth re-formulation, given respectively by

$$(NS) \;\; \min\{F(\mathbf{x}) \mid \mathbf{x} \in \overline{\mathbf{S}}\} \qquad \Longleftrightarrow \qquad (SF) \;\; \min_{\mathbf{x}, \mathbf{w}} \left\{\mathbf{C}_1(\mathbf{x}, \mathbf{w}) \mid w^i \in \Delta^i\right\}$$

where

$$F(\mathbf{x}) = \sum_{i=1}^{m} \nu_i \min_{1\le l\le k} d(x^l, a^i); \qquad C_1(\mathbf{x},\mathbf{w}) := \sum_{i=1}^{m}\sum_{l=1}^{k} \nu_i w_l^i d(x^l, a^i),$$

and with $\mathbf{w} := (w^1, \ldots w^m) \in \Delta = \Delta^1 \times \ldots \times \Delta^m$.

6.1 The Fuzzy k-Means Algorithm (FKM)

This method (Bezdek, 1981) was originally devised to relax the solution of problem (SF), by introducing the objective function

$$\mathbf{C}_\beta(\mathbf{x},\mathbf{w}) := \sum_{i=1}^{m}\sum_{l=1}^{k} \nu_i (w_l^i)^\beta d(x^l, a^i),$$

where $\beta > 1$ is the parameter that governs the "fuzzy partition" through $\mathbf{w}$. Indeed, the *nonlinearity* of the function $\mathbf{w} \to \mathbf{C}_\beta(\mathbf{x},\mathbf{w})$, (as opposed to the standard k-means objective which corresponds to $\beta = 1$) yields a solution $\mathbf{w}$ which is not anymore of the binary type as in hard clustering, hence the "fuzzy" terminology, (which also corresponds to the *soft* terminology). Applying the Gauss-Seidel algorithm described in the appendix to problem (SF) with the objective $\mathbf{C}_\beta(\mathbf{x},\mathbf{w})$ yields the FKM, (see, for example, Duda et al., 2001, page. 528).

Alternatively, keeping $\mathbf{x}$ fixed, and minimizing with respect to $\mathbf{w}$, that is, solving the strictly convex problem in $\mathbf{w}$:

$$\mathbf{w}^*(\mathbf{x}) = \operatorname{argmin}\{\mathbf{C}_\beta(\mathbf{x},\mathbf{w}) \mid w^i \in \Delta^i, i = 1,\ldots,m\},$$

one obtains the optimal solution

$$(w_l^i)^*(\mathbf{x}) = d(x^l,a^i)^{\frac{1}{1-\beta}} \left(\sum_{j=1}^{k} d(x^j,a^i)^{\frac{1}{1-\beta}}\right)^{-1}, \quad i = 1,\ldots,m,\ l = 1,\ldots k.$$

Plugging-in the optimal solution $\mathbf{w}^*(\mathbf{x})$ into the objective $\mathbf{C}_\beta(\mathbf{x},\mathbf{w})$, an easy computation shows that the remaining optimization problem to be solved in the variable $\mathbf{x}$ reduces to:

$$\min\left\{\sum_{i=1}^{m} \nu_i \left(\sum_{l=1}^{k} d(x^l,a^i)^{\frac{1}{1-\beta}}\right)^{1-\beta} \mid \mathbf{x} \in \overline{\mathbf{S}}\right\},\ (\beta > 1). \tag{39}$$

Therefore, with the choice $h(t) = (-t)^{1/(1-\beta)}$ (which is in the class $\mathcal{H}$, see Example 12) in the definition of $F_s(\cdot)$ as given in (15), one obtains that with the particular choice $\pi_l = k^{-1}, \forall l$, the objective function (39), and hence the resulting FKM algorithm, are recovered as a special case of the smoothing approach and of algorithm **SKM**, for which our convergence result applies for any distance $d \in \mathcal{D}(S)$, thus also broadening the scope of FKM based method. Note that with the special choice $\beta = 2$, the Harmonic Mean algorithm (cf. Example 10) is recovered.

6.2 The Deterministic Annealing (DA)

In Rose et al. (1990), building on statistical physics analogies, the authors have introduced the Deterministic Annealing (DA) algorithm for clustering problems. In the recent work (Teboulle and Kogan, 2005) we already announced that DA can be derived and interpreted as a smoothing optimization method. Indeed, the DA algorithm simply corresponds to the choice $\mathcal{H} \ni h(t) = e^t$, in the nonlinear mean G_h, and when substituted in (15) yields to solve the smooth nonconvex optimization problem:

$$\min\left\{-s\sum_{i=1}^{m}\nu_i \log\sum_{l=1}^{k}\pi_l e^{-d(x^l,a^i)/s} \mid \mathbf{x}\in\overline{\mathbf{S}}\right\}, \tag{40}$$

where the smoothing parameter $s > 0$ plays the role of the inverse temperature used in the DA formulation (Rose et al., 1990). Thus, applying **SKM** to problem (40), we obtain the classical DA algorithm whenever $d(\cdot,\cdot)$ is the usual squared Euclidean distance, as well as its extension with $d \in \mathcal{D}(S)$.

The DA algorithm is thus also a smoothing optimization method. It has been claimed in the literature (Rose et al., 1990; Ueda and Nakano, 1998; Rose, 1998), but, to the best of our knowledge, not mathematically proven, that by suitably tuning the temperature, namely in our language, the smoothing parameter, the DA can deliver "global" optimal solutions. However, our current analysis demonstrates that only the nonsmoothness difficulty appears to be eliminated via the DA approach, yet the nonconvexity difficulty remains. Nevertheless, deterministic annealing based algorithms continue to be successfully used in practice (Elkan, 2006) and appear to share two particularly interesting and unique features:
(i) As reported in several studies, the DA converges very quickly to"good" solutions (as compared to the k-means algorithm).
(ii) The DA algorithm which is obtained from our framework with the special choice $h(t) = e^t$, is also a source of many other seemingly different methods, in particular when we consider its extension with distance-like functions other than the usual squared Euclidean distance.

In view of the combined smoothing and successive approximation approach we have developed, these two features are perhaps not too surprising in the following sense. The quick delivery of a reasonable approximate solution relies on the gradient descent property of the **SKM** algorithm developed in Section 5, and hence of the DA algorithm in particular. Moreover, the Log-Sum exponential function appears to be *optimal* in the sense we previously explained in Section 4 and in Remark 13. Below, we further exemplify the point (ii) by briefly showing how the methods mentioned in the introduction, such as, Maximum Entropy Clustering Algorithms (MECA), Expectation Maximization (EM), and the Bregman soft clustering algorithm are essentially equivalent smoothing methods.

6.3 Deterministic Annealing, Entropy Methods and Information Theory

A remarkable mathematical property of the Log-Sum exponential function is that it is just the conjugate of the entropy function on the unit simplex, and vice-versa. More precisely, the following result holds.

Lemma 18 *For any given* $\pi \in \Delta$,

$$\log\sum_{l=1}^{k}\pi_l e^{z_l} = \max_{y\in\Delta}\left\{\langle y,z\rangle - \sum_{l=1}^{k} y_l \log\frac{y_l}{\pi_l}\right\},$$

where $\Delta = \{y \in \mathbb{R}^k : \sum_{l=1}^{k} y_l = 1, y \geq 0\}$. *Moreover, with* $g(z) = \log\sum_{l=1}^{k}\pi_l e^{z_l}$, *one has*

$$g^*(y) = \sum_{l=1}^{k} y_l \log\frac{y_l}{\pi_l}, \text{ with } \operatorname{dom} g^* = \Delta.$$

Proof. By direct computation, or see for example Rockafellar (1970, p. 148). ■

Using the dual representation of the Log-Sum exponential function given in Lemma 18 into the objective function of (40), some algebra shows that the smooth optimization problem (40) is equivalent to:

$$\min_{\mathbf{x},\mathbf{w}}\left\{\mathbf{C}_1(\mathbf{x},\mathbf{w})+s\sum_{i=1}^{m}\sum_{l=1}^{k}\nu_i w_l^i\log\frac{w_l^i}{\pi_l}\mid w^i\in\Delta^i,\ i=1,\ldots,m\right\}. \tag{41}$$

This equivalent reformulation of the smooth optimization model (40) allows to show connections with other approaches that we now discuss.

Problem (40) recovers the basic formulation of what is called in the literature *Maximum Entropy Clustering Algorithms*, (MECA) (Rose, 1998). Of course, this shows that maximum entropy methods applied to the clustering problem, are thus a special case of our smoothing approach.

Furthermore, it is interesting to notice that in MECA models one usually assume that $\pi_l = k^{-1}, \forall l = 1,\ldots k.$, that is, a uniform distribution. We can enrich the model by considering π_l as weights (probabilities) associated to each cluster center l, and ask to find the "best" possible distribution for π, namely for given $(\mathbf{x},\mathbf{w})$ in problem (41), we need to solve:

$$\min\left\{-s\sum_{i=1}\nu_i\sum_{l=1}^{k}w_l^i\log\pi_l\mid\pi\in\Delta_+\right\}.$$

Clearly, the objective function in the later problem is convex in π, and a straightforward application of Karush-Khun-Tucker (KKT) optimality conditions (Bertsekas, 99) to the latter problem yields the optimal choice for π:

$$\pi_l^* = \sum_{i=1}^{m}\nu_i w_l^i,\ l = 1,\ldots k. \tag{42}$$

Another interpretation is to view MECA as follows. Going back to the formulation (SF), the problem (41) can in fact be viewed from various angles via the classical penalty-barrier optimization method, which is also a smoothing approach (Auslender, 1999), whereby the entropy is used to penalize the simplex constraints on w^i, and s would play the role of the penalty-barrier parameter. Namely by defining,

$$E(\mathbf{w},\pi) := \sum_{i=1}^{m}\sum_{l=1}^{k}\nu_i w_l^i\log\frac{w_l^i}{\pi_l}$$

with $\mathbf{w}\in\Delta$, where $\mathbf{w} := (w^1,\ldots w^m)$, $w^i\in\mathbb{R}^k$, and $\Delta = \Delta^1\times\ldots\times\Delta^m$, problem (41) can be viewed as a family of penalized problems, with $s > 0$, being the penalty parameter for solving the constrained problem:

$$\min_{\mathbf{x},\mathbf{w}}\{C_1(\mathbf{x},\mathbf{w})\mid E(\mathbf{w},\pi)\le\varepsilon,\mathbf{w}\in\Delta\},$$

where $\varepsilon > 0$, is preassigned. An interesting interpretation of the latter problem was described by Rose (1998) as follows. The smoothing parameter s can be viewed as a Lagrange multiplier to an entropy constraint which would measure the level of randomness in the following sense: the first term in the objective of (41) is a predefined "expected distortion", and thus we are trading "entropy", the second term in (41), for reduction in the distortion as $s\to 0$. Alternatively, problem (41) can also be seen directly related to the fundamental Shannon rate-distortion function (Berger, 1971). Suppose the knowledge of the "expected distortion" $\mathbf{C}_1(\mathbf{x},\mathbf{w})$ is pre-assumed at a certain level, say $\delta > 0$, that is, one has $C_1(\mathbf{x},\mathbf{w}) = \sum_{i,l} \nu_i w^i_l d(x^l, a^i) \leq \delta$. Fix any $s > 0$, say $s = 1$, and let π be as given in (42). Then, for a fixed given $\mathbf{x}$ one has to solve,

$$R(\delta) = \min_{\mathbf{w}} \{E(\mathbf{w},\pi) \mid \mathbf{C}_1(\mathbf{x},\mathbf{w}) \leq \delta,\ \mathbf{w} \in \Delta\}. \tag{43}$$

Following information theory concepts (Thomas and Cover, 1991), a close inspection of the last problem reveals that the objective function $E(\mathbf{w},\pi)$ in (43) is the so-called average mutual information functional, and the optimal value $R(\delta)$ of problem (43) is nothing else but the mathematical description of the rate distortion function. The later problem is a convex optimization problem in $\mathbf{w}$ (a probabilistic/soft assignment variable), that can be solved via the so-called Blahut-Arimoto algorithm, (Berger, 1971), which is an iterative fixed point type convex dual optimization method.

6.4 The EM algorithm and Bregman Soft Clustering

The Expectation Maximization (EM) algorithm is a workhorse in statistical estimation problems for learning mixtures of distributions, (see, for example, Duda et al., 2001). In a very recent paper, Banerjee et al. (2005) have shown that there exists a bijective correspondence between regular exponential distributions and Bregman divergences. This result enables them to show (see, Banerjee et al., 2005, Section 5), that the EM algorithm for learning mixtures of regular exponential family distributions is in fact equivalent to Bregman soft clustering.

Without recourse to any probability/statistical arguments, we provide below, yet another interpretation and realization of this result, by showing that it corresponds to a special case of our generic scheme, with the special choices $h(t) = e^t$, and $d(x,a) := d_\psi(a,x)$.

Fix any $s > 0$, say $s = 1$ in (40). Then, one has to solve the equivalent problem (after an obvious change of sign to pass to maximization):

$$\max_{\mathbf{x}} \mathcal{F}(x,\pi) \equiv \max\left\{\sum_{i=1}^{m} \nu_i \log \sum_{l=1}^{k} \pi_l e^{-d(x^l,a^i)} \mid \mathbf{x} \in \mathbf{S}\right\},$$

where $\pi \in \Delta_+$. Applying **SKM**, given $\pi \in \Delta_+$, and $\bar{x}^l \in \mathbf{S}$ we first need to compute:
The E-Step: compute "conditional probabilities" (cf. (18), and 22)):

$$\rho^{il}(\bar{\mathbf{x}}) := \frac{\pi_l e^{-d(\bar{x}^l,a^i)}}{\sum_{j=1}^{k} \pi_j e^{-d(\bar{x}^l,a^i)}}. \tag{44}$$

Now, the second step in **SKM** consists of solving

$$\min_{\mathbf{x}\in\mathbf{S}} \left\{\sum_{i=1}^{m}\sum_{l=1}^{k} \nu_i d(x^l,a^i)\rho^{il}(\bar{\mathbf{x}})\right\},$$

which admits a unique global minimizer (cf. Example 3), and yields
The M-step:

$$x^l = \frac{\sum_{i=1}^{m} \nu_i \rho^{il}(\bar{\mathbf{x}}) a^i}{\sum_{j=1}^{m} \nu_j \rho^{jl}(\bar{\mathbf{x}})},\ l = 1,\ldots,k. \tag{45}$$

Now, if we assume that π^l is also considered as a variable in the maximization of $\mathcal{F}(x,\pi)$, (e.g., π_l gives the fraction of points representing optimal clusters l), then given $\bar{\mathbf{x}} \in \mathbf{S}$, one has to solve

$$\max_{\pi} \mathcal{F}(x,\pi) \equiv \max \left\{ \sum_{i=1}^{m} \nu_i \log \sum_{l=1}^{k} \pi_l e^{-d(\bar{x}^l, a^i)} \mid \pi \in \Delta_+ \right\}. \tag{46}$$

It is easy to see that the objective function in problem (46) is a concave function in the variable π, and hence a direct application of the KKT optimality conditions to this convex problem (maximizing a concave objective subject to a simplex constraint) yields using the notations in (44), the global optimal solution:

$$\pi_l^* = \sum_{i=1}^{m} \nu_i \rho^{il}(\bar{\mathbf{x}}),\ l = 1,\dots,k. \tag{47}$$

Therefore, with the equations (44), (45) and (47), we have recovered through a special realization of **SKM**, the EM algorithm for learning mixtures model of exponential family distributions or equivalently the Bregman soft clustering method, as recently derived in Banerjee et al. (2005) from a completely different perspective.

6.5 Discussion

There exists many other related clustering algorithms not discussed here, that can be designed, analyzed and extended through our framework, and we refer the reader to the relevant cited references throughout this paper and their bibliography therein. The above comparisons were just briefly outlined to demonstrate the fundamental and useful role that convex analysis and optimization theory can play in the analysis and interpretation of iterative center-based clustering algorithms. As such, the general framework we have proposed should be viewed as complementary to alternative formulations and approaches. Indeed, it is also important to mention that for specific application domains which often involve particular input data representation, such as in statistics and information theory, alternative approaches and formulations should not be ignored, as they can at times provide ways for better/new insights or/and solution methods. For example, in the problem of learning mixture models, an alternative approach is via spectral projection techniques, which provide algorithms with theoretical guarantee for learning mixtures of log-concave distributions (Kannan et al., 2005). Another example is the information bottleneck method (Tishby et al., 1999) which provides a useful formalism and principle to extract relevant information in a given data set. The recent interesting study (Banerjee et al., 2005) connecting Bregman clustering and lossy compression schemes through an information theoretic formalism, and which allows for extending the information bottleneck method, further demonstrates the usefulness of considering alternative formulations.

7. Concluding Remarks

This paper is a theoretical contribution to clustering analysis, and has three messages. First, the proposed optimization framework and formalism provides a systematic and simple way to design and analyze a broad class of hard and soft center-based clustering algorithms, which retain the computational simplicity of the k-means algorithm. Secondly, the proposed formalism has provided a closure and unification to a long list of disparate motivations and approaches that have been proposed for center-based clustering methods. As discussed in the paper, many of these algorithms which have been widely used in applications are special cases of our analysis. Third, the common optimization language and the fundamental tools we have used, which rely on the combination of convex asymptotic functions, nonlinear means and distance-like functions, and from which our generic scheme has emerged, enables for a rigorous analysis of center-based clustering algorithms, have revealed the advantages and limitations of such methods, and have provided the basis for significantly extend the scope of partitioning clustering algorithms.

As a final remark, we hope that the current study will further stimulate the use and application of convex analysis and optimization in data analysis. Indeed, there are several theoretical and practical challenges that need to be met in future research works in clustering analysis. Let us mention just a few questions that naturally emerged from the present analysis. As already pointed out throughout this study, for a given specific data set to cluster, current experimental results indicate that with the choice of the Euclidean squared distance, the deterministic annealing algorithm (based on the log-exponential mean) and the harmonic k-means can produce better quality clustering (see, for example, Rose, 1998; Ueda and Nakano, 1998; Zhang et al., 1999; Hamerly and Elkan, 2002; Elkan, 2006). Thus, future experimentation based on these methods, but using other proximity measures that could model various data types, deserves to be considered. Furthermore, as pointed out by a referee, it would be interesting to identify the "best" choice of the function h to be used in the broader family of convex nonlinear means. Similarly, can we characterize the classes of functions h or/and the classes distance-like functions d that would allow us to eliminate or/and control the inherent nonconvexity difficulty which is present in the clustering problem? Can we rigorously measure the quality of clustering produced by the generic scheme, or some other possible refined variants, in terms of the problem data and the couple (h,d)? Even partial answers to such theoretical questions would have a significant practical impact, and deserve further investigations.

Acknowledgments

We thank an anonymous referee for his constructive comments that prompted us to streamline our results, and have helped to greatly improve the presentation of the paper. We also thank the area editor Charles Elkan for his valuable comments and for pointing out additional relevant references. This research was partially supported by the United States-Israel Binational Science Foundation under BSF grant 2002-010.

Appendix A.

In this appendix we briefly describe the basic mechanism of hard clustering center-based algorithms.

A.1 The Gauss-Seidel Algorithm-GSA

The Gauss-Seidel algorithm, also called coordinate descent or alternative optimization method, proceeds as follows to solve the generic minimization problem:

$$\min\min\left\{F(x,y) : x \in X, y \in Y\right\}.$$

- At iteration t, fix $x(t) \in X$, and minimize with respect to y the function $F(x(t),y)$, to get $y(t)$.
- Update x by minimizing $F(x,y(t))$ with respect to x.

- Continue this process iteratively until some declared stopping criteria is satisfied.

Convergence of GSA to a stationary point can be established under suitable conditions on the problem's data, (Auslender, 1976; Bertsekas, 99).

A.2 Hard Clustering with Distance-Like Functions

The popular k-means algorithm with d being the squared of the Euclidean distance, is nothing else but the Gauss-Seidel method, when applied to the exact smooth formulation (ES) of the clustering problem.

$$\min_{\mathbf{x},\mathbf{w}} \left\{ \mathbf{C}_1(\mathbf{x},\mathbf{w}) = \sum_{i=1}^{m} \sum_{l=1}^{k} \nu_i w_l^i d(x^l, a^i) \right\}.$$

More generally, applying GSA on problem (ES) with distance-like functions d, we can obtain a broad class of general Hard Clustering algorithms **HCD**. Note that the main computational step in the generic **HCD** algorithm keeps the computational simplicity of the k-means algorithm, yet allows for significantly expand the scope of hard clustering center-based methods.
Algorithm HCD–Hard Clustering with Distance-Like Functions

- **Step 0-Initialization** Set $t = 0$ and let $\{x^l(0) : l = 1, \dots, k\}$ be the k initial centers in S. (These can be picked randomly).

- **Step 1-Cluster Assignment** For $i = 1, \dots, m$ solve $w^i(t) = \operatorname{argmin}_{w \in \Delta_i} \sum_{l=1}^{k} w_l d(x^l(t), a^i)$.

- **Step 2-Update Cluster Centers** For each $l = 1, \dots, k$ solve

$$\left(x^1(t+1), \dots, x^k(t+1)\right) = \operatorname*{argmin}_{x^1,\dots,x^k} \left\{ \sum_{i=1}^{m} \sum_{l=1}^{k} \nu_i w_l^i(t) d(x^l, a^i) \right\}.$$

- **Step 3-Stopping Criteria** Stop when some stopping criteria is satisfied, (e.g., $\mathbf{x}(t+1) = \mathbf{x}(t)$), else set $t \longleftarrow t+1$ and goto **Step 1**.

Algorithm HCD clearly implies that the objective function of (ES) is nonincreasing at the successive iterations. The resulting stationary point obtained by this procedure satisfies the Karush-Khun-Tucker (KKT) necessary optimality conditions for problem (ES) (Bertsekas, 99).

The remarkable simplicity of algorithm HCD relies on the fact that Step 1 is trivially solved, while step 2 can be solved *analytically* for a wide class of distances d. Indeed, at any given iteration t, to solve Step 1, for all $i = 1, \dots, m$, let $l(i) = \operatorname*{argmin}_{1 \le l \le k} d(x^l(t), a^i)$. Then, an optimal w^i is simply given by

$$w_{l(i)}^i(t) = 1, \text{ that is, when } x^l \text{ is the center closest to } a^i, \text{and } w_l^i(t) = 0, \forall l \neq l(i).$$

To solve step 2, noting that the objective is *separable* in each variable x^l, it reduces to solve for each x^l:

$$x^l(t+1) = \operatorname*{argmin}_{x} \left\{ \sum_{i=1}^{m} w_{l(i)}^i(t) d(x, a^i) \right\}.$$

For the class $d \in \mathcal{D}(S)$, as well as for other distance-like functions as discussed in Section 2.2, this problem admits a unique global optimal solution. Furthermore, for distance-like functions which are given separable, this problem even reduces to solve a *one dimensional* optimization problem, which can often be solved analytically for many examples (Teboulle et al., 2006). It is easy to see that algorithm **HCD** includes as special cases, not only the popular k-means algorithm, but also many others hard clustering methods mentioned in the introduction. In particular, it includes and extend the Bregman hard clustering algorithm recently derived in Banerjee et al. (2005, Algorithm 1, page 1715).

References

A. Auslender. *Optimisation: Methodes Numériques*. Masson, Paris, 1976.

A. Auslender. Penalty and barrier methods: a unified framework. *SIAM J. Optimization* 10(1), 211-230, 1999.

A. Auslender and M. Teboulle. *Asymptotic Cones and Functions in Optimization and Variational Inequalities*. Springer–Verlag, New York, 2003.

A. Auslender and M. Teboulle. Interior gradient and proximal methods for convex and conic optimization. *SIAM J. Optimization*, 16(3):697-725, 2006.

A. M. Bagirov, A. M. Rubinov, N.V. Soukhoroukova, and J. Yearwood. Unsupervised and supervised data classification via nonsmooth and global optimization. *TOP*, (Formerly Trabajos Investigación Operativa) 11(1):1–93, 2003.

A. M. Bagirov and J. Ugon. An algorithm for minimizing clustering functions. *Optimization*, 54(4-5): 351–368, 2005.

A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh. Clustering with Bregman Divergences. *Journal of Machine Learning Research*, 6, 1705–1749, 2005.

A. Ben-Tal and M. Teboulle. Expected utility, penalty functions, and duality in stochastic nonlinear programming. *Management Sciences*, 32(11):1445–1466, 1986.

A. Ben-Tal and M. Teboulle. A smoothing technique for nondifferentiable optimization problems. In *Springer Verlag Lecture Notes in Mathematics*, volume 1405, pages 1–11, Berlin, 1989.

A. Ben-Tal, M. Teboulle, and W. H. Yang, A least-squares based method for a class of nonsmooth minimization problems with applications in plasticity. *Applied Mathematics and Optimization*, 24(3):273–288, 1991.

T. Berger. *Rate Distortion Theory: A Mathematical Basis for Data Compression*. Prentice Hall, Englewood Cliffs, New Jersey, 1971.

D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, Belmont, Massachusetts, second edition, 1996.

D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, Massachusetts, second edition, 1999.

D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, New Jersey, 1989.

J. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York, 1981.

L. M. Bregman. A relaxation method of finding a common point of convex sets and its application to the solution of problems in convex programming. *USSR Comp. Math. and Math Phys.*,7:200-217, 1967.

J. Brimberg and Love R.F. Global convergence of a generalized iterative procedure for the minisum location problem with l_p distances. *Operations Research*, 41:1153–1163, 1993.

Y. Censor and A. Lent. An interval row action method for interval convex programming. *J. of Optimization Theory and Applications*, 34:321–353, 1981.

Y. Censor and S. A. Zenios, *Parallel Optimization*, Oxford University Press, Oxford, 1997.

G. Chen and M. Teboulle. Convergence analysis of a proximal-like algorithm using Bregman functions. *SIAM J. on Optimization* 3:538–543, 1993.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 1991.

I. Csiszar. Information-type measures of difference of probability distributions and indirect observations. *Studia Sci. Mathematica Hungarica*, 2:299–318, 1967.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, Inc., second edition, 2001.

C. Elkan. Clustering documents with an exponential-family approximation of the Dirichlet compound multinomial distribution. In *Proceedings of the 23rd International Conference on Machine Learning ICML 06*, 289-296, 2006.

E. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, Abstract, 21(3):768, 1965.

M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman and Company, San Francisco, CA, 1979.

A.D. Gordon and J.T. Henderson. An algorithm for Euclidean sum of squares classification. *Biometrics*, 33:355-362, 1977.

R.M. Gray and D.L. Neuhoff. Quantization. *IEEE Transaction on Information Theory*, 44(6):2325–2382, 1998.

G. Hamerly and C. Elkan. Alternatives to the k-means algorithm that find better clusterings. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management (CIKM'02)*, pages 600-607, 2002.

J P. Hansen and N. Mladenovic. J-Means: A New Heuristic for Minimum Sum-of-Squares Clustering. *Pattern Recognition* 34:405–413, 2001.

G. Hardy, J.E. Littlewood, and G. Polya. *Inequalities*. Cambridge University Press, Cambridge, 1934.

A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering: A review, *ACM Computing Surveys*, 31(3):264-323, 1999.

R. Kannan, H. Salmasian, and S. Vempala. The spectral method for general mixture models. In *Proceedings of the 18th Annual Conference on Learning Theory*, 444-457, 2005.

N.B. Karayiannis. An axiomatic approach to soft learning vector quantization and clustering. *IEEE Transactions on Neural Networks*, 10(5):1153-1165, 1999.

S.P. Lloyd. Least squares quantization in PCM. Bell Telephone Laboratories Paper, Murray Hill, NJ, 1957. Also in, *IEEE Transactions on Information Theory*, 28:127-135, 1982.

Y. Linde, A. Buzo, and R.M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1):84-95, 1980.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley Symposium on Math., Stat. and Probability*, pages 281–296, 1967.

D. Modha and S. Spangler. Feature weighting in k-means clustering. *Machine Learning*, 52(3):217-237, 2003.

M.R. Rao. Cluster analysis and mathematical programming. *J. American Statistical Association*, 66:622–626, 1971.

R. T. Rockafellar. *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.

K. Rose, E. Gurewitz, and C.G. Fox. A deterministic annealing approach to clustering. *Pattern Recognition Letters*, 11(9):589–594, 1990.

K. Rose. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proceedings of the IEEE*, 86(11):2210–2239, 1998.

H. Steinhaus. Sur la division des corps materiels en parties. *Bull. Acad. Polon. Sci.*, C1. III, vol. IV, 801–804, 1956.

M. Teboulle. “On φ-divergence and its applications”. In *Systems and Management Science by Extremal Methods* (F.Y. Phillips, J. Rousseau, eds.), Kluwer Academic Press, chap. 17, pages 255–273, 1992.

M. Teboulle. Entropic proximal mappings with application to nonlinear programming. *Mathematics of Operations Research*, 17:670–690, 1992.

M. Teboulle. Convergence of proximal-like algorithms. *SIAM J. of Optimization*, 7:1069-1083, 1997.

M. Teboulle and J. Kogan. Deterministic annealing and a k-means type smoothing optimization algorithm. In *Proceedings of the Workshop on Clustering High Dimensional Data and its Applications* (held in conjunction with the Fifth SIAM International Conference on Data Mining). I. Dhillon, J. Ghosh and J. Kogan (eds.), pages 13-22, 2005.

M. Teboulle, P. Berkhin, I. Dhillon, Y. Guan, and J. Kogan. Clustering with entropy-like k-means algorithms. In *Grouping Multidimensional Data: Recent Advances in Clustering*. J. Kogan, C. Nicholas, and M. Teboulle, (Eds.), Springer Verlag, NY, pages 127–160, 2006.

N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck method. In *Proc. of the 37th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

N. Ueda and R. Nakano. Deterministic annealing EM algorithm. *Neural Networks*, 11(2): 271–282, 1998.

E. Weiszfeld. Sur le point pour lequel la somme des distances de *n* points donnés est minimum. *Tohoku Mathematical Journal*, 43:355–386, 1937.

B. Zhang, M. Hsu, and U. Dayal. K-harmonic means - a data clustering algorithm. Technical Report HPL-1999-124 991029, HP Labs, Palo Alto, CA, 1999.

Journal of Machine Learning Research 8 (2007) 103-130 Submitted 3/05; Revised 7/06; Published 1/07

Minimax Regret Classifier for Imprecise Class Distributions

Rocío Alaiz-Rodríguez ROCIO.ALAIZ@UNILEON.ES
Dpto. de Ingeniería Eléctrica y de Sistemas y Automática
Campus de Vegazana, Universidad de León
24071 León, Spain

Alicia Guerrero-Curieses ALICIA.GUERRERO@URJC.ES
Dpto. de Teoría de la Señal y Comunicaciones
Campus de Fuenlabrada, Universidad Rey Juan Carlos
Camino del Molino s/n, 28943 Fuenlabrada-Madrid, Spain

Jesús Cid-Sueiro JCID@TSC.UC3M.ES
Dpto. de Tecnologías de las Comunicaciones
EPS, Universidad Carlos III de Madrid
Avda. de la Universidad, 30, 28919 Leganés-Madrid, Spain

Editor: Dale Schuurmans

Abstract

The design of a minimum risk classifier based on data usually stems from the stationarity assumption that the conditions during training and test are the same: the misclassification costs assumed during training must be in agreement with real costs, and the same statistical process must have generated both training and test data. Unfortunately, in real world applications, these assumptions may not hold. This paper deals with the problem of training a classifier when prior probabilities cannot be reliably induced from training data. Some strategies based on optimizing the worst possible case (conventional minimax) have been proposed previously in the literature, but they may achieve a robust classification at the expense of a severe performance degradation. In this paper we propose a *minimax regret* (*minimax deviation*) approach, that seeks to minimize the maximum deviation from the performance of the optimal risk classifier. A neural-based *minimax regret* classifier for general multi-class decision problems is presented. Experimental results show its robustness and the advantages in relation to other approaches.

Keywords: classification, imprecise class distribution, minimax regret, minimax deviation, neural networks

1. Introduction - Problem Motivation

In the general framework of learning from examples and specifically when dealing with uncertainty, the robustness of the decision machine becomes a key issue. Most machine learning algorithms are based on the assumption that the classifier will use data drawn from the same distribution as the training data set. Unfortunately, for most practical applications (such as remote sensing, direct marketing, fraud detection, information filtering, medical diagnosis or intrusion detection) the target class distribution may not be accurately known during learning: for example, because the cost of labelling data may be class-dependent or the prior probabilities are non-stationary. Therefore, the data used to design the classifier (within the Bayesian context (see VanTrees, 1968), the prior probabilities and the misclassification costs) may be non representative of the underlying real distributions.

If the ratio of training data corresponding to each class is not in agreement with real class distributions, designing Bayes decision rules based on prior probabilities estimated from these data will be suboptimal and can seriously affect the reliability and performance of the classifier.

A similar problem may arise if real misclassification costs are unknown during training. However, they are usually known by the end user, who can adapt the classifier decision rules to cost changes without re-training the classifier. For this reason, our attention in this paper is mainly focused on the problem of uncertainty in prior probabilities. Furthermore, being aware that class distribution is seldom known (at least totally) in real world applications, a robust approach (as opposite to adaptive) that prevents severe performance degradation appears to be convenient for these situations.

Besides other adaptive and robust approaches that address this problem (discussed in more detail in Section 2.2) it is important to highlight those that handle the problem of uncertainty in priors by following a robust minimax principle: minimize the maximum possible risk. Analytic foundations of minimax classification are widely considered in the literature (see VanTrees, 1968; Moon and Stirling, 2000; Duda et al., 2001, for instance) and a few algorithms to carry out minimax decisions have been proposed. From computationally expensive ones such as estimating probability density functions (Takimoto and Warmuth, 2000; Kim, 1996) or using methods from optimization (Polak, 1997) to simpler ones like neural network training algorithms (Guerrero-Curieses et al., 2004; Alaiz-Rodriguez et al., 2005).

Minimax classifiers may, however, be seen as over-conservative since its goal is to optimize the performance under the least favorable conditions. Consider, for instance, a direct marketing campaign application carried out in order to maximize profits. Since optimal decisions rely on the proportion of potential buyers and it is usually unknown in advance, our classification system should take into account this uncertainty. Nevertheless, following a pure minimax strategy can lead to solutions where minimizing the maximum loss implies considering there are no potential clients. If it is the case, this minimax approach does not seem to be suitable for this kind of situation.

In this imprecise class distribution scenario, it can be noticed that the classifier performance may be highly deviated from the optimal, that is, that of the classifier knowing actual priors. Minimizing this gap (that is, the maximum possible deviation with respect to the optimal classifier) is the focus of this paper. We seek for a system as robust as the conventional minimax approach but less pessimistic at the same time. We will refer to it as a *minimax deviation* (or *minimax regret*) classifier. In contrast to other robust and adaptive approaches, it can be used in general multiclass problems. Furthermore, as shown in Guerrero-Curieses et al. (2004), minimax approaches can be used in combination with the adaptive proposal by Saerens et al. (2002) to exploit its advantages.

This *minimax regret* approach has recently been applied in the context of parameter estimation (Eldar et al., 2004; Eldar and Merhav, 2004) and a similar competitive strategy has been used in the context of hypothesis testing (Feder and Merhav, 2002).

Under prior uncertainty, our solution provides an upper bound of the performance divergence from the optimal classifier. We propose a simple learning rate scaling algorithm in order to train a neural-based *minimax deviation* classifier. Although training can be based on minimizing any objective function, we have chosen objective functions that provide estimates of the posterior probabilities (see Cid-Sueiro and Figueiras-Vidal, 2001, for more details).

This paper is organized as follows: the next section provides an overview of the problem as well as some previous approaches to cope with it. Next, Section 3 states the fundamentals of minimax classification together with a deeper analysis of the *minimax regret* approach proposed in this paper. Section 4 presents a neural training algorithm to get a neural-based *minimax regret* classifier under complete uncertainty. Moreover, practical situations with partial uncertainty in priors are also discussed. A learning algorithm to solve them is provided in Section 5. In Section 6, some experimental results show that *minimax regret* classifiers outperform (in terms of maximum risk deviation) classifiers trained on re-balanced data sets and those with the originally assumed priors. Finally, the main conclusions are summarized in Section 7.

2. Problem Overview

Traditionally, supervised learning lies in the fact that training data and real data come from the same (although unknown) statistical model. In order to carefully analyze to what extend classifier performance depends on conditions such as class distribution or decision costs, learning and decision theory principles are briefly revisited. Next, some previous approaches to deal with environment imprecision are reviewed.

2.1 Learning and Making Optimal Decisions

Let $S = \{(\mathbf{x}^k, \mathbf{d}^k), k = 1, \ldots, K\}$ denote a set of labelled samples where $\mathbf{x}^k \in \mathbb{R}^N$ is an observation feature vector and $\mathbf{d}^k \in U_L = \{\mathbf{u}_0, \ldots, \mathbf{u}_{L-1}\}$ is the label vector. Class-i label $\mathbf{u}_i$ is a unit L-dimensional vector with components $u_{i,j} = \delta_{ij}$, with every component equal to 0, except the i-th component which is equal to 1.

We assume a learning process that estimates parameters $\mathbf{w}$ of a non-linear mapping $\mathbf{f}_\mathbf{w} : \mathbb{R}^N \to \mathcal{P}$ from the input space into probability space $\mathcal{P} = \{\mathbf{p} \in [0,1]^L | \sum_{i=0}^{L-1} p_i = 1\}$. The *soft* decision is given by $\mathbf{y}^k = \mathbf{f}_\mathbf{w}(\mathbf{x}^k) \in \mathcal{P}$ and the hard output of the classifier is denoted by $\widehat{\mathbf{d}}$. Note that $\mathbf{d}$ and $\widehat{\mathbf{d}}$ will be used to distinguish the actual class from the predicted one, respectively.

Several costs (or benefits) associated with each possible decision are also defined: c_{ij} denotes the cost of deciding in favor of class i when the true class is j. Negative values represent benefits (for instance, c_{ii}, which is the cost of correctly classifying a sample from class i could be negative in some practical cases).

In general cost-sensitive classification problems, either misclassification costs c_{ij} or c_{ii} costs can take different values for each class. Thus, there are many applications where classification errors lead to very different consequences (medical diagnosis, fault detection, credit risk analysis), what implies misclassification costs c_{ij} that may largely vary between them. In the same way, there are also many domains where correct decision costs (or benefits) c_{ii} do not take the same value. For instance, in targeted marketing applications (Zadrozny and Elkan, 2001), correctly identifying a buyer implies some benefit while correctly classifying a non buyer means no income. The same applies to medical diagnosis domains such as the gastric carcinoma problem studied in Güvenir et al. (2004). In this case, the benefit of correct classification also depends on the class: the benefit of correctly classifying an early stage tumor is higher than that of a later stage.

The expected risk (or loss) R is given by

$$R = \sum_{j=0}^{L-1} \sum_{i=0}^{L-1} c_{ij} P\{\widehat{\mathbf{d}} = \mathbf{u}_i | \mathbf{d} = \mathbf{u}_j\} P_j \ , \tag{1}$$

where $P\{\widehat{\mathbf{d}} = \mathbf{u}_i | \mathbf{d} = \mathbf{u}_j\}$ with $i \neq j$ represent conditional error probabilities, and $P_j = P\{\mathbf{d} = \mathbf{u}_j\}$ is the prior probability of class $\mathbf{u}_j$.

Defining the conditional risk of misclassifying samples from class $\mathbf{u}_j$ as

$$R_j = \sum_{i=0}^{L-1} c_{ij} P\{\widehat{\mathbf{d}} = \mathbf{u}_i | \mathbf{d} = \mathbf{u}_j\} \ ,$$

we can express risk (1) as

$$R = \sum_{i=0}^{L-1} R_i P_i \ . \tag{2}$$

It is well-known that the Bayes decision rule for the minimum risk is given by

$$\widehat{\mathbf{d}} = \arg \min_{\mathbf{u}_i} \{ \sum_{j=0}^{L-1} c_{ij} P\{\mathbf{d} = \mathbf{u}_j | \mathbf{x}\}\} \ , \tag{3}$$

where $P\{\mathbf{d} = \mathbf{u}_i | \mathbf{x}\}$ is the a *posteriori* probability of class i given sample $\mathbf{x}$.

The optimal decision rule depends on posterior probabilities and therefore, on the prior probabilities and the likelihood.

In theory, as long as posterior probabilities (or likelihood and prior probabilities) are known, the optimal decision in Eq. (3) can be expressed after a trivial manipulation as a function of the cost differences between the costs $(c_{ij} - c_{jj})$ (Duda et al., 2001). This is the reason why c_{jj} is usually assumed to be zero and the value of the cost difference is directly assigned to c_{ij}. When dealing with practical applications, however, some authors (Zadrozny and Elkan, 2001; Güvenir et al., 2004) have urged to use meaningful decision costs measured over a common baseline (and not necessarily taking $c_{jj} = 0$) in order to avoid mistakes that otherwise could be overlooked. For this reason and, what is more important, the uncertainty class distribution problem addressed in this paper, decision costs measured over a common baseline are considered. Furthermore, absolute values of decision costs are relevant to the design of classifiers under the minimax regret approach.

2.2 Related Work: Dealing with Cost and Prior Uncertainty

Most proposals to address uncertainty in priors fall into the categories of adaptive and robust solutions. While the aim of a robust solution is to avoid a classifier with very poor performance under any conditions, an adaptive system pursues to fit the classifier parameters using more incoming data or more precise information.

With an adaptive-oriented principle, Provost (2000) states that, once the classifier is trained under specific class distributions and cost assumptions (not necessarily the operating conditions), the selection of the optimal classifier for specific conditions is carried out by a correct placement of the decision thresholds. In the same way, the approaches in Kelly et al. (1999) and Kubat et al. (1998) consider that tuning the classifier parameters should be left to the end user, expecting that class distributions and misclassification costs will be precisely known then.

Some graphical methods based on the ROC curve have been proposed in Adams and Hand (1998) and Provost and Fawcett (2001) in order to compare the classifier performance under imprecise class distributions and/or misclassification costs. The ROC convex hull method presented in Provost and Fawcett (2001) (or the alternative representation proposed in Drummond and Holte (2000)) allows the user to select potentially optimal classifiers, providing a flexible way to select them when precise information about priors or costs is available. Under imprecision, some classifiers can be discarded but this does not necessarily provide a method to select the optimal classifier between the possible ones and fit its parameters. Furthermore, due to its graphical character, these methods are limited to binary classification problems.

Changes in prior probabilities have also been discussed by Saerens et al. (2002), who proposes a method based on re-estimating the prior probabilities of real data in an unsupervised way and subsequently adjusting the outputs of the classifier according to the new a *priori* probabilities. Obviously, the method requires enough unlabelled data being available for re-estimation.

As an alternative to adaptive schemes, several robust solutions have been proposed, as the resampling methods, especially in domains where imbalanced classes come out (Kubat and Matwin, 1997; Lawrence et al., 1998; Chawla et al., 2002; Barandela et al., 2003). Either by undersampling or oversampling, the common purpose is to balance artificially the training data set in order to get a uniform class distribution, which is supposed to be the least biased towards any class and, thus, the most robust against changes in class distributions.

The same approach is followed in cost sensitive domains, but with some subtle differences in practice. It is well known that class priors and decision costs are intrinsically related. For instance, different decision costs can be simulated by altering the priors and vice versa (see Ting, 2002, for instance). Thus, when a uniform distribution is desired in a cost sensitive domain, but working with cost insensitive decision machines, class priors are altered according to decision costs, what is commonly referred as *rebalancing*.

The manipulation of the training data distribution has been applied to cost-sensitive learning in two-class problems (Breiman et al., 1984) in the following way: basically, the class with higher misclassification cost (suppose n times the lowest misclassification cost) is represented with n times more examples than the other class. Besides random sampling strategies, other sampling-based *rebalancing* schemes have been proposed to accomplish this task, like those considering closeness to the boundaries between classes (Japkowicz and Stephen, 2002; Zhou and LiuJ, 2006) or the cost-proportionate rejection sampling presented in Zadrozny et al. (2003). Extending the formulation of this type of procedures to general multiclass problems with multiple (and possibly asymmetric) inter-class misclassification costs appears to be a nontrivial task (Zadrozny et al., 2003; Zhou and LiuJ, 2006), but some progress has been made recently with regard to this latter point (Abe et al., 2004). Note, also, that many (although not all) of these rebalancing strategies are usually implemented by oversampling and/or subsampling, that is, replicating examples (without adding any extra information) and/or deleting them (which implies information loss).

3. Robust Classifiers Under Prior Uncertainty: Minimax Classifiers

Prior probability uncertainty can be coped from a robust point of view following a minimax derived strategy. Minimax regret criterion is discussed in this section after presenting the conventional minimax criterion.

Although our approach extends to general multi-class problems and the discussion is carried out in that way, we will first illustrate, for the sake of clarity and simplicity, a binary situation.

3.1 Minimax Classifiers

As Eq. (3) shows, the minimum risk decisions depend on the misclassification costs, c_{ij}, and the posterior class probabilities and, thus, they depend on the prior probabilities, P_i. Different prior distributions (frequency for each class) give rise to different Bayes classifiers. Fig. 1 shows the Bayes risk curve, $R_B(P_1)$ versus class-1 prior probability for a binary classification problem.

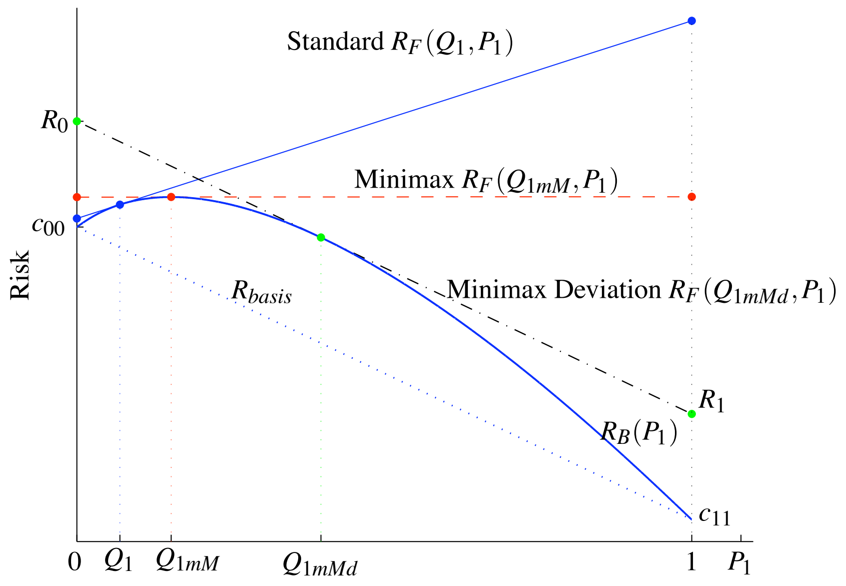


Figure 1: Risk vs. P_1. Minimum risk curve and performance under prior changes for the *standard*, *minimax* and *minimax deviation* classifier. $R_B(P_1)$ stands for the optimal Bayes Risk against P_1. $R_F(Q_1,P_1)$ denotes the Risk of a standard classifier (Fixed decision rule optimized for prior probabilities Q_1 estimated in the training phase) against P_1. $R_F(Q_{1mM},P_1)$ denotes the Risk of a *minimax* classifier (Fixed decision rule optimized for the minimax probabilities Q_{1mM}) against P_1. $R_F(Q_{1mMd},P_1)$ denotes the Risk of a *minimax deviation* classifier (Fixed decision rule optimized for the minimax deviation probabilities Q_{1mMd}) against P_1.

If the prior probability distribution is unknown when the classifier is designed, or this distribution changes with time or from one environment to other, the mismatch between training and test conditions can degrade significantly the classifier performance.

For instance, assume that $\mathbf{Q} = (Q_0, Q_1)$ is the vector with class-0 and class-1 prior probabilities estimated in the training phase, respectively, and let $R_B(Q_1)$ represent the minimum (Bayes) risk attainable by any decision rule for these priors. Note, that, according to Eq. (2), for a given classifier, the risk is a linear function of priors. Thus, risk $R_F(Q_1,P_1)$ associated to the (fixed) classifier optimized for $\mathbf{Q}$ changes linearly with actual prior probabilities P_1 and $P_0 = 1 - P_1$, going from $(0,R_0)$ to $(1,R_1)$ (the continuous line in Fig. 1), where R_0 and R_1 refer to the class conditional risks for classes 0 and 1, respectively. Fig. 1 shows the impact of this change in priors and how performance deviates from optimal.

Also, it can be shown (see VanTrees, 1968, for instance) that the minimum risk curve obtained for each prior is convex and the risk function of a given classifier verifies $R_F(Q_1,P_1) \geq R_B(P_1)$ with a tangent point at $P_1 = Q_1$.

The dashed line in Fig. 1 shows the performance of the *minimax* classifier, which minimizes the maximum possible risk under the least favorable priors, thus providing the most robust solution, in the sense that performance becomes independent from priors. From Fig. 1, it becomes clear that the minimax classifier is optimal for prior probabilities $\mathbf{P} = \mathbf{Q_{mM}} = (Q_{0mM}, Q_{1mM})$ maximizing R_B. Thus, this strategy is equivalent to maximizing the minimum risk (Moon and Stirling, 2000; Duda et al., 2001). We will refer to them as the minimax probabilities.

Fig. 1 also makes clear that although a minimax classifier is a robust solution to address the imprecision in priors, it may become a somewhat pessimistic approach.

3.2 Minimax Deviation Classifiers

We propose an alternative classifier that, instead of minimizing the maximum risk, minimizes the maximum deviation (regret) from the optimal Bayes classifier. In the following, we will refer to it as the *minimax deviation* or *minimax regret* classifier.

A comparison between *minimax* and *minimax deviation* approaches is also shown in Fig. 1. This latter case corresponds to a classifier trained on prior probabilities $\mathbf{P} = \mathbf{Q_{mMd}}$ with performance as a function of priors given by a line (a plane or hyperplane for three or more classes, respectively) parallel to what we name, in the following, basis risk ($R_{basis} = c_{00}(1-P_1) + c_{11}P_1$).

Note that the maximum deviation (with respect to priors) of the classifier optimized for $\mathbf{Q}$ is given by

$$D(\mathbf{Q}) = \max_{P_1}\{R_F(Q_1,P_1) - R_B(P_1)\} = \max\{R_0 - c_{00}, R_1 - c_{11}\} \quad .$$

The inspection of Fig. 1 shows that the minimum of D (with respect to $\mathbf{Q}$) is achieved when

$$R_0 - c_{00} = R_1 - c_{11} \quad ,$$

which means that line $R_F(Q_1,P_1)$ is parallel to arc named R_{basis} in the figure and tangent to R_B at Q_{1mMd}. Therefore, the *minimax regret* classifier is also the Bayes solution with respect to the least favorable priors (Q_{0mMd}, Q_{1mMd}) (see Berger, 1985, for instance), which will be denoted as minimax deviation probabilities.

Now, we extend the formulation to a general L-class problem.

Definition 1 *Consider a L-class decision problem with costs $c_{ij}, 0 \le i, j < L$ and $c_{jj} \le c_{ij}$, and let $R_{\mathbf{w}}(\mathbf{P})$ be the risk of a decision machine with parameter vector* $\mathbf{w}$ *when prior class probabilities are given by $\mathbf{P} = (P_0, \dots, P_{L-1})$. The deviation function is defined as*

$$D_{\mathbf{w}}(\mathbf{P}) = R_{\mathbf{w}}(\mathbf{P}) - R_B(\mathbf{P})$$

and the minimax deviation is defined as

$$D_{mMd} = \inf_{\mathbf{w}} \max_{\mathbf{P}} \{D_{\mathbf{w}}(\mathbf{P})\} \quad . \tag{4}$$

Note that the above definition assumes that the maximum exists. This is actually the case, since $D_{\mathbf{w}}(\mathbf{P})$ is a linear function over a compact set, $\mathcal{P}$. Note, also, that our definition includes the natural assumption that c_{jj} is never higher than c_{ij}, meaning that making a decision error is always less costly than taking the correct decision. This assumption is used in part of our theoretical analysis.

The algorithms proposed in this paper are based on the fact that the minimax deviation can be computed without knowing R_B

Theorem 2 *The minimax deviation is given by*

$$D_{mMd} = \inf_{\mathbf{w}} \max_{\mathbf{P}} \{\overline{D}_{\mathbf{w}}(\mathbf{P})\} \ ,$$

where

$$\overline{D}_{\mathbf{w}}(\mathbf{P}) = R_{\mathbf{w}}(\mathbf{P}) - R_{basis}(\mathbf{P}) \tag{5}$$

and

$$R_{basis}(\mathbf{P}) = \sum_{j=0}^{L-1} c_{jj} P_j \ . \tag{6}$$

Proof Note that, according to Eqs. (1) and (2), for any decision machine and any $\mathbf{u}_i \in \mathcal{U}_L$,

$$R(\mathbf{u}_j) = R_j = \sum_{i=0}^{L-1} c_{ij} P\{\widehat{\mathbf{d}} = \mathbf{u}_i | \mathbf{d} = \mathbf{u}_j\} \geq c_{jj} \ .$$

Since the bound is reached by the classifier deciding $\widehat{\mathbf{d}} = \mathbf{u}_j$ for any observation $\mathbf{x}$, we have $R_B(\mathbf{u}_j) = c_{jj}$. Therefore, using Eq. (6), we find that, for any $\mathbf{u} \in \mathcal{U}_L$,

$$R_B(\mathbf{u}) = R_{basis}(\mathbf{u})$$

and, thus,

$$D_{\mathbf{w}}(\mathbf{u}) = \overline{D}_{\mathbf{w}}(\mathbf{u}) \ .$$

Since Bayes minimum risk $R_B(\mathbf{P})$ is a convex function of priors and $R_{\mathbf{w}}(\mathbf{P})$ is linear, $D_{\mathbf{w}}(\mathbf{P})$ is concave and, thus, it is maximum at some of the vertices in $\mathcal{P}$ (i.e., at some $\mathbf{P} = \mathbf{u} \in \mathcal{U}_L$). Thus,

$$\max_{\mathbf{P}} \{D_{\mathbf{w}}(\mathbf{P})\} = \max_{\mathbf{u} \in \mathcal{U}_L} \{D_{\mathbf{w}}(\mathbf{u})\} \ . \tag{7}$$

Since the maximum difference between two hyperplanes defined over $\mathcal{P}$ is always at some vertex, we can conclude that

$$\max_{\mathbf{P}} \{\overline{D}_{\mathbf{w}}(\mathbf{P})\} = \max_{\mathbf{u} \in \mathcal{U}_L} \{\overline{D}_{\mathbf{w}}(\mathbf{u})\} = \max_{\mathbf{u} \in \mathcal{U}_L} \{D_{\mathbf{w}}(\mathbf{u})\} \ . \tag{8}$$

Combining Eqs. (4), (7) and (8), we get

$$D_{mMd} = \inf_{\mathbf{w}} \max_{\mathbf{P}} \{\overline{D}_{\mathbf{w}}(\mathbf{P})\} \ .$$

■

Note that R_{basis} represents the risk baseline of the ideal classifier with zero errors. Th. 2 shows that the minimax regret can be computed as the minimax deviation to this ideal classifier. Note, also, that if costs c_{ii} do not depend on i, Eq. (5) becomes equivalent (up to a constant) to the Bayes risk and the minimax regret classifier becomes equivalent to the minimax classifier .

Another important result for the algorithms proposed in this paper is that, under some conditions on the minimum risk, the minimum and maximum operators can be permuted. Although general results on the permutability of minimum and maximum operators can be found in the literature (see Polak, 1997, for instance), we provide here the proof for the specific case interesting to this paper.

Theorem 3 *Consider the minimum deviation function given by*

$$\overline{D}_{\min}(\mathbf{P}) = \inf_{\mathbf{w}}\{\overline{D}_{\mathbf{w}}(\mathbf{P})\} \ , \tag{9}$$

where $\overline{D}_{\mathbf{w}}(\mathbf{P})$ is the normalized deviation function given by Eq. (5), and let $\mathbf{P}^*$ be the prior probability vector providing the maximum deviation,

$$\mathbf{P}^* = \arg\max_{\mathbf{P}}\left\{\overline{D}_{\min}(\mathbf{P})\right\} \ . \tag{10}$$

If $\overline{D}_{\min}(\mathbf{P})$ is continuously differentiable at $\mathbf{P} = \mathbf{P}^*$, then the minimax deviation, D_{mMd}, defined by Eq. (4), is

$$D_{mMd} = \overline{D}_{\min}(\mathbf{P}^*) = \max_{\mathbf{P}}\inf_{\mathbf{w}}\left\{\overline{D}_{\mathbf{w}}(\mathbf{P})\right\} \ . \tag{11}$$

Proof

For any classifier with parameter vector $\mathbf{w}$, we can write,

$$\max_{\mathbf{P}}\overline{D}_{\mathbf{w}}(\mathbf{P}) \geq \overline{D}_{\mathbf{w}}(\mathbf{P}^*) \geq \overline{D}_{\min}(\mathbf{P}^*)$$

and, thus,

$$\inf_{\mathbf{w}}\max_{\mathbf{P}}\overline{D}_{\mathbf{w}}(\mathbf{P}) \geq \overline{D}_{\min}(\mathbf{P}^*) \ . \tag{12}$$

Therefore, $\overline{D}_{\min}(\mathbf{P}^*)$ is a lower bound of the minimax regret.

Now we prove that $\overline{D}_{\min}(\mathbf{P}^*)$ is also an upper bound. According to Eq. (9), for any $\varepsilon > 0$, there exists a parameter vector $\mathbf{w}_\varepsilon$ such that

$$\overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P}^*) \leq \overline{D}_{\min}(\mathbf{P}^*) + \varepsilon \ . \tag{13}$$

By definition, for any $\mathbf{P}, \overline{D}_{\min}(\mathbf{P}) \leq \overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P})$. Therefore, using Eq. (13), we can write

$$\overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P}^*) - \overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P}) \leq \overline{D}_{\min}(\mathbf{P}^*) - \overline{D}_{\min}(\mathbf{P}) + \varepsilon \ . \tag{14}$$

Since $\overline{D}_{\min}(\mathbf{P})$ is continuously differentiable and (according to Eq. (10)) maximum at $\mathbf{P}^*$, for any $\varepsilon' > 0$ there exists $\delta > 0$ such that, for any $\mathbf{P} \in \mathcal{P}$ with $\|\mathbf{P}^* - \mathbf{P}\| \leq \delta$ we have

$$\overline{D}_{\min}(\mathbf{P}^*) - \overline{D}_{\min}(\mathbf{P}) \leq \varepsilon'\|\mathbf{P}^* - \mathbf{P}\| \leq \varepsilon'\delta \ . \tag{15}$$

Let $\mathbf{P}_\delta$ a prior such that $\|\mathbf{P}^* - \mathbf{P}_\delta\| = \delta$. Taking $\varepsilon = \varepsilon'\delta$ and combining Eqs. (14) and (15) we can write

$$\overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P}^*) - \overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P}_\delta) \leq 2\varepsilon'\delta \ .$$

Since the above condition is verified for any $\varepsilon' > 0$ and any prior $\mathbf{P}_\delta$ at distance δ from $\mathbf{P}$, and taking into account that $\overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P})$ is a linear function of $\mathbf{P}$, we conclude that the maximum slope of $\overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P})$ is bounded by $2\varepsilon'$ and, thus, for any $\mathbf{P} \in \mathcal{P}$, we have

$$\overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P}) - \overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P}^*) \leq 2\varepsilon'\|\mathbf{P} - \mathbf{P}^*\| \leq 2\sqrt{2}\varepsilon' \ ,$$

(where we have used the fact that the maximum distance between two probability vectors is $\sqrt{2}$). Therefore, we can write

$$\max_{\mathbf{P}}\overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P}) \leq \overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P}^*) + 2\sqrt{2}\varepsilon'$$

and, thus,

$$\inf_{\mathbf{w}} \max_{\mathbf{P}} \overline{D}_{\mathbf{w}}(\mathbf{P}) \leq \overline{D}_{\mathbf{w}_\varepsilon}(\mathbf{P}^*) + 2\sqrt{2}\varepsilon' \ .$$

Finally, using Eq. (13) and taking into account that $\varepsilon = \varepsilon'\delta \leq \sqrt{2}\varepsilon'$ we get

$$\inf_{\mathbf{w}} \max_{\mathbf{P}} \overline{D}_{\mathbf{w}}(\mathbf{P}) \leq \overline{D}_{\min}(\mathbf{P}^*) + 3\sqrt{2}\varepsilon' \ . \tag{16}$$

Since the above is true for any $\varepsilon' > 0$ we conclude that $\overline{D}_{\min}(\mathbf{P}^*)$ is also an upper bound of $\overline{D}_{\mathbf{w}}$. Therefore, combining Eqs. (12) and (16), we conclude that

$$\inf_{\mathbf{w}} \max_{\mathbf{P}} \overline{D}_{\mathbf{w}}(\mathbf{P}) = \overline{D}_{\min}(\mathbf{P}^*) \ ,$$

which completes the proof. ■

Note that the deviation function needs to be neither differentiable nor a continuous function of **w** parameters.

If the minimum deviation function is not continuously differentiable at the minimax deviation probability, $\mathbf{P}^*$, the theorem cannot be applied. The reason is that, although there should exist at least one classifier providing the minimum deviation at $\mathbf{P} = \mathbf{P}^*$, it or they could not provide a constant deviation with respect to the prior probability. The situation can be illustrated with an example.

Let $x \in \mathbb{R}$ be given by $p(x|d=0) = 0.8N(x,\sigma) + 0.2N(x-2,\sigma)$ and $p(x|d=1) = 0.2N(x-1,\sigma) + 0.8N(x-3,\sigma)$, where $\sigma = 0.5$ and $N(x,\sigma) = (2\pi\sigma)^{-1/2}\exp(-x^2/(2\sigma^2))$, and consider the set Φ_λ of classifiers given by a single threshold over x and decision

$$\hat{d} = \begin{cases} 1 & \text{if } x \geq \lambda \\ 0 & \text{if } x < \lambda. \end{cases}$$

Fig. 2 shows the distribution of both classes over x, and Fig. 3 shows, as a function of priors, the minimum error probability (continuous line) that can be obtained using classifiers in Φ_λ. Note that decision costs $c_{00} = c_{11} = 0$ and $c_{01} = c_{10} = 1$ have been considered for this illustrative problem. An abrupt slope change is observed at the minimax deviation probability, for $P\{d=1\} = 1/2$. For this prior, there are two single threshold classifiers providing the minimum error probability, which are given by thresholds λ_1 and λ_2 in Fig. 2. However, as shown in Fig. 3 neither of them provides a risk that is constant in the prior. The minimax deviation classifier in Φ_λ, which has a threshold λ_0, does not attain minimum risk at the minimax deviation probability and, thus, cannot be obtained by using Eq. (11).

For this example, the desired robust classifier should have a deviation function given by the horizontal dotted line in Fig. 3. Fortunately, it can be obtained by combining the outputs of several classifiers. For instance, let $\hat{d}_1$ and $\hat{d}_2$ the decisions of classifiers given by thresholds λ_1 and λ_2, respectively. It is not difficult to see that the classifier selecting $\hat{d}_1$ and $\hat{d}_2$ at random (for each input sample x) provides a robust classifier.

This procedure can be extended to the multiclass-case: consider a set of L classifiers with parameters $\mathbf{w}_k$, $k = 0, \ldots, L-1$, and consider the classifier such that, for any input sample x, makes a decision equal to $\widehat{d_k}$ (i.e., the decision of classifier with parameters $\mathbf{w}_k$), with probability q_k. It is not difficult to show that the deviation function of this classifier is given by

$$\overline{D}(\mathbf{P}) = \sum_{j=0}^{L-1} P_j \left(\sum_{k=0}^{L-1} q_k \overline{D}_j(\mathbf{w}_k) \right) \ ,$$

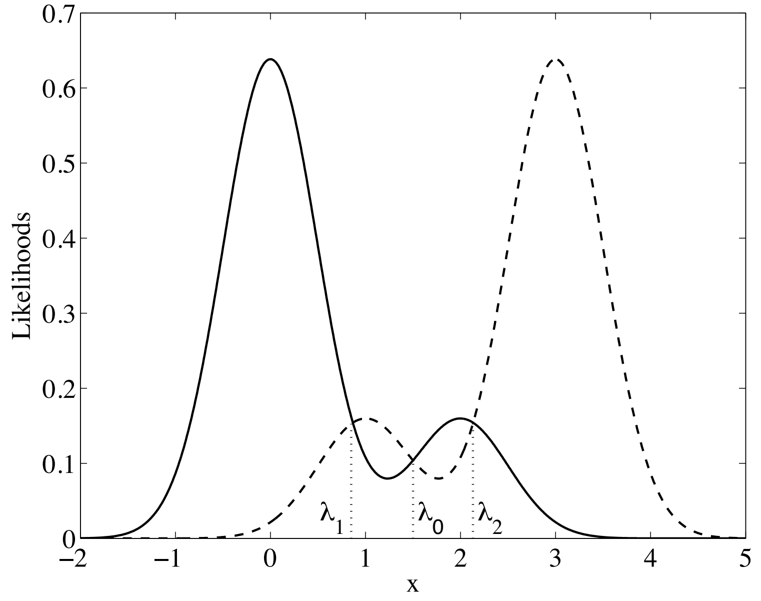


Figure 2: The conditional data distributions for the one-dimensional example discussed in the text. λ_1 and λ_2 are the thresholds providing the minimum risk at the minimax deviation probability. λ_0 provides the minimax deviation classifier.

where $\overline{D}_j(\mathbf{w}_k) = R_j(\mathbf{w}_k) - c_{jj}$. In order to get a constant deviation function, probabilities q_k should be chosen in such a way that

$$\sum_{k=0}^{L-1} q_k \overline{D}_j(\mathbf{w}_k) = D \quad ,$$

where D is a constant. Solving these linear equations for q_k, $k = 0, \ldots, L-1$ (with the constraint $\sum_k q_k = 1$), the required probabilities can be found.

Note that, in order to build the non-deterministic classifier providing a constant deviation, a set of L independent classifiers that are optimal at the minimax deviation prior should be found. However, we go no further on the investigation of this special case for two main reasons:

- The situation does not seem to be common in practice. In our simulations, we have found that the maximum of the minimum risk deviation always provided a response which is approximately parallel to R_{basis}.
- In general, the abrupt change in the derivative may be a symptom that the classifier structure is not optimal for the data distribution. Instead of building a nondeterministic classifier, increasing the classifier complexity should be more efficient.

Although the least favorable prior providing the minimax deviation can be computed in closed form for some simple distributions, in general, it must be computed numerically. Moreover, we assume here that the data distribution is not known, and must be learned from examples. Thus,

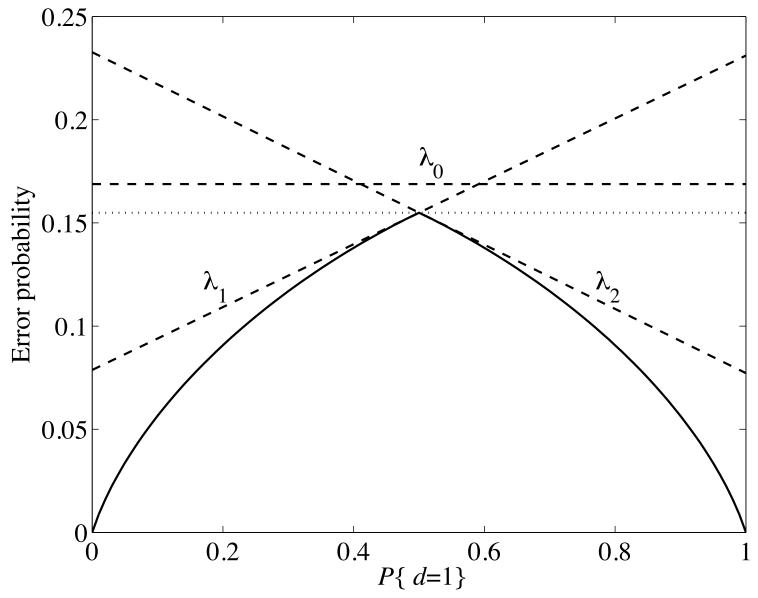


Figure 3: Error probabilities as a function of prior probability of class 1 for the example in Fig. 2. Thresholds λ_1 and λ_2 do not provide the minimax deviation classifier, which is obtained for threshold λ_0. However, the random combination of classifiers with thresholds λ_1 and λ_2 (dotted line) provides a robust classifier with deviation lower than that of λ_0.

we must incorporate the estimation of the least favorable prior in the learning process. Next, we propose a training algorithm in order to get a *minimax regret* classifier based on neural networks.

4. Neural Robust Classifiers Under Complete Uncertainty

Note that, if $\mathbf{Q}_{mMd}$ is the probability vector providing the maximum in Eq. (11), that is,

$$\mathbf{Q}_{mMd} = \arg\max_{\mathbf{P}} \left\{ \inf_{\mathbf{w}} \{ \overline{D}_{\mathbf{w}}(\mathbf{P}) \} \right\} ,$$

then we can write

$$D_{mMd} = \inf_{\mathbf{w}} \{ \overline{D}_{\mathbf{w}}(\mathbf{Q}_{mMd}) \} .$$

Therefore, the *minimax deviation* classifier can be estimated by training a classifier using prior in $\mathbf{Q}_{mMd}$. For this reason, $\mathbf{Q}_{mMd}$ will be called the *minimax deviation* prior (or least favorable prior). Our proposed algorithms are based on an iterative process of estimating parameters $\mathbf{w}$ based on an estimate of the minimax deviation prior, and re-estimating prior based on an estimate of network weights. This is shown in the following.

4.1 Updating Network Weights

Learning is based on minimizing some empirical estimate of the overall error function

$$E\{C(\mathbf{y},\mathbf{d})\} = \sum_{i=0}^{L-1} P\{\mathbf{d}=\mathbf{u}_i\}E\{C(\mathbf{y},\mathbf{d})|\mathbf{d}=\mathbf{u}_i\} = \sum_{i=0}^{L-1} P_i C_i \ ,$$

where $C(\mathbf{y},\mathbf{d})$ may be any error function and C_i is the expected conditional error for class-i.

Selecting the appropriate error function (see Cid-Sueiro and Figueiras-Vidal, 2001, for instance), learning rules can be designed providing *a posteriori* probability estimates ($y_i \approx P\{\mathbf{d} = \mathbf{u}_i|\mathbf{x}\}$, where y_i is the soft decision) and, thus, according to Eq. (3), the hard decision minimizing the risk can be approximated by

$$\widehat{\mathbf{d}} = \arg\min_i \{ \sum_{j=0}^{L-1} c_{ij} y_j \} \ .$$

The overall empirical error function (cost function) used in learning for priors $\widehat{\mathbf{P}} = (\widehat{P}_0, \ldots, \widehat{P}_{L-1})$ may be written as

$$\begin{aligned}
\widehat{C} &= \sum_{i=0}^{L-1} \widehat{P}_i \widehat{C}_i = \sum_{i=0}^{L-1} \widehat{P}_i \frac{1}{K_i} \sum_{k=1}^{K} d_i^k \widehat{C}(\mathbf{y}^k,\mathbf{d}^k), \\
&= \frac{1}{K}\left[\sum_{i=0}^{L-1}\left(\frac{\widehat{P}_i}{K_i/K}\sum_{k=1}^{K} d_i^k C(\mathbf{y}^k,\mathbf{d}^k)\right)\right], \\
&= \frac{1}{K}\sum_{k=1}^{K}\left[\sum_{i=0}^{L-1}\frac{\widehat{P}_i}{\widehat{P}_i^{(0)}} d_i^k \widehat{C}(\mathbf{y}^k,\mathbf{d}^k)\right] \ ,
\end{aligned} \tag{17}$$

where $\widehat{P}_i^{(0)} = K_i/K$ is an initial estimate of class-i prior based on class frequencies in the training set and $\widehat{P}_i$ is the current prior estimate.

Minimizing error function (17) by means of a stochastic gradient descent learning rule leads to update the network weights at k-th iteration as

$$\begin{aligned}
\mathbf{w}^{(k+1)} &= \mathbf{w}^{(k)} - \mu\Big(\sum_{i=0}^{L-1}\frac{\widehat{P}_i^{(n)}}{\widehat{P}_i^{(0)}} d_i^k \nabla_{\mathbf{w}} C(\mathbf{y}^k,\mathbf{d}^k)\Big), \\
&= \mathbf{w}^{(k)} - \Big(\sum_{i=0}^{L-1}\mu_i^{(n)} d_i^k\Big)\nabla_{\mathbf{w}} C(\mathbf{y}^k,\mathbf{d}^k) \ ,
\end{aligned} \tag{18}$$

where

$$\mu_i^{(n)} = \mu\frac{\widehat{P}_i^{(n)}}{\widehat{P}_i^{(0)}} \tag{19}$$

is a learning step scaled by the prior ratio. Note that d_i selects the appropriate $\mu_i^{(n)}$ according to the pattern class membership. The classifier is trained without altering the original training data set class distribution $\widehat{P}_i^{(0)}$ and therefore, without missing or duplicating information.

4.2 Updating Prior Probabilities

Eq. (11) shows that the learning process should maximize (5) with respect to the prior probabilities. The estimate of (5) can be computed as

$$\widehat{\bar{D}}_{\mathbf{w}}(\mathbf{P}) = \widehat{R}_{\mathbf{w}}(\mathbf{P}) - R_{basis}(\mathbf{P}) \ , \tag{20}$$

where

$$\widehat{R}_{\mathbf{w}}(\mathbf{P}) = \sum_{j=0}^{L-1} \widehat{R}_j P_j \tag{21}$$

is the overall Bayes risk estimate and

$$\widehat{R}_j = \frac{1}{N_j} \sum_{i=0}^{L-1} c_{ij} N_{ij} \tag{22}$$

is the class-j conditional risk estimate where N_j is the number of class $\mathbf{u}_j$ patterns in the training phase and N_{ij} is the number of samples from class $\mathbf{u}_j$ assigned to $\mathbf{u}_i$.

In order to derive a learning rule to find an estimate $\widehat{P}_i$ satisfying constraints $\sum_{i=0}^{L-1} \widehat{P}_i = 1$ and $0 \le \widehat{P}_i \le 1$, we will use auxiliary variables B_i such that

$$\widehat{P}_i = \frac{\exp(B_i)}{\sum_{j=0}^{L-1} \exp(B_j)} \ . \tag{23}$$

We maximize $\widehat{\bar{D}}_{\mathbf{w}}$ with respect to B_i. Applying the chain rule,

$$\frac{\partial \widehat{\bar{D}}_{\mathbf{w}}}{\partial B_i} = \sum_{j=0}^{L-1} \frac{\partial \widehat{\bar{D}}_{\mathbf{w}}}{\partial \widehat{P}_j} \frac{\partial \widehat{P}_j}{\partial B_i} \ ,$$

and using Eqs. (20), (21) and (23), we get

$$\begin{aligned}
\frac{\partial \widehat{\bar{D}}_{\mathbf{w}}}{\partial B_i} &= \sum_{j=0}^{L-1} (\widehat{R}_j - c_{jj}) \widehat{P}_i (\delta_{ij} - \widehat{P}_j), \\
&= \widehat{P}_i \left(\widehat{R}_i - c_{ii} - \sum_{j=0}^{L-1} (\widehat{R}_j \widehat{P}_j) + \sum_{j=0}^{L-1} (c_{jj} \widehat{P}_j) \right), \\
&= \widehat{P}_i \left(\left(\widehat{R}_i - c_{ii} \right) - \left(\widehat{R}_{\mathbf{w}} - \widehat{R}_{basis} \right) \right), \\
&= \widehat{P}_i \widehat{R}_{di} \ ,
\end{aligned}$$

where

$$\widehat{R}_{di} = (\widehat{R}_i - c_{ii}) - (\widehat{R}_{\mathbf{w}} - \widehat{R}_{basis}) \ .$$

The learning rule for auxiliary variable B_i is

$$\begin{aligned}
B_i^{(n+1)} &= B_i^{(n)} + \rho \frac{\partial \widehat{D}_{\mathbf{w}}}{\partial B_i}, \\
&= B_i^{(n)} + \rho \widehat{P}_i^{(n)} \widehat{R}_{di}^{(n)} \ ,
\end{aligned} \tag{24}$$

where parameter $\rho > 0$ controls the rate of convergence. Using Eq. (23) and Eq. (24), the updated learning rule for $\widehat{P}_i$ is

$$\begin{aligned}\widehat{P}_i^{(n+1)} &= \frac{\exp(B_i^{(n)})\exp\left(\rho\widehat{P}_i^{(n)}\widehat{R}_{di}^{(n)}\right)}{\sum_{j=0}^{L-1}\left[\exp\left(B_j^{(n)}\right)\exp\left(\rho\widehat{P}_j^{(n)}\widehat{R}_{dj}^{(n)}\right)\right]}, \\ &= \frac{\widehat{P}_i^{(n)}\exp\left(\rho\widehat{P}_i^{(n)}\widehat{R}_{di}^{(n)}\right)}{\sum_{j=0}^{L-1}\left[\widehat{P}_j^{(n)}\exp\left(\rho\widehat{P}_j^{(n)}\widehat{R}_{dj}^{(n)}\right)\right]} . \end{aligned} \tag{25}$$

4.3 Training Algorithm for a Minimax Deviation Classifier

In the previous section, both the network weights updating rule (18) and the prior probability update rule (25) have been derived. The algorithm resulting from the combination is shown as follows:

```
for $n = 0$ to $N_{iterations} - 1$ do
  for $k = 1$ to $K$ do
    $\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \Big(\sum_{i=0}^{L-1} \mu_i^{(n)} d_i^k\Big)\nabla_{\mathbf{w}} C(\mathbf{y}^k, \mathbf{d}^k)$
  end for
  Estimate $\widehat{R}^{(n)}, \widehat{R}_i^{(n)}, i = 0, \ldots, L-1$, according to (21) and (22)
  Update minimax probability $\widehat{P}_i^{(n+1)}, i = 0, \ldots, L-1$ according to (25) and compute $\mu_i^{(n+1)}$ with (19)
end for
```

5. Robust Classifiers Under Partial Uncertainty

Although in many practical situations prior probabilities may not be specified with precision, they can be partially known. In this section we discuss how partial information about priors can be used to improve the classifier performance in relation to a complete uncertainty situation.

From now on, let us consider that lower (or upper) bounds of the priors are known based on previous experience. We will denote the lower and upper bounds of class-i prior probability as P_{il} and P_{iu}, respectively.

In order to illustrate this situation consider a binary classification problem where probability lower bounds P_{0l} and P_{1l} are known. That is, $P_1 \in [P_{1l}, 1 - P_{0l}]$ where this interval represents the uncertainty region. Let us denote by $\Gamma = \{\mathbf{P} : 0 \leq P_i \leq 1, \sum_{i=0}^{L-1} P_i = 1, P_i \geq P_{il}\}$ the probability region satisfying the imposed constraints. In the following, we will refer to Γ as the *uncertainty region*.

Now, the aim is to design a classifier that minimizes the maximum regret from the minimum risk only inside the uncertainty region. This is depicted in Fig. 4(a), which shows that reducing the uncertainty in priors allows to reduce deviation from the optimal classifier. This minimax regret approach for the uncertainty region Γ is often called Γ-minimax regret. As discussed before, the minimax deviation solution gives a Bayes solution with respect some priors denoted in the partial uncertainty case as $\mathbf{Q}_{mMd}^{\Gamma}$ in Fig. 4(a), which is the least favorable distribution according to the regret criterion.

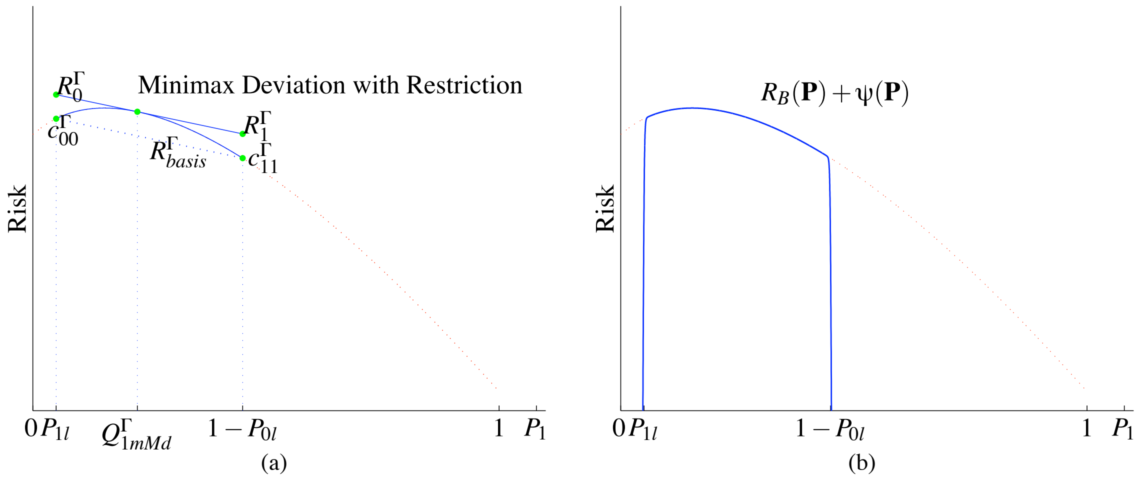


Figure 4: Minimax deviation classifier under partial uncertainty of prior probabilities: (a)Γ-*minMaxDev* Classifier. (b) Modified cost function defined as $R_B(\mathbf{P})+\psi(\mathbf{P})$.

In contrast to the minimax regret criterion, note that a classical minimax classifier for the considered uncertainty region would minimize the worst-case risk. It would be a Bayes solution for the prior where the minimum risk reaches its maximum and it could be denoted as $\mathbf{Q}_{mM}^\Gamma$.

Notice, also, that these solutions will be the same if the risk for the vertex of Γ take the same value ($c_{ii}^\Gamma = k$).

5.1 Neural Robust Classifiers Under Partial Uncertainty

Minimax search can be formulated as maximizing (with respect to priors) the minimum (with respect to network parameters) of deviation function (5), as described in previous section, but subject to some constraints

$$\begin{array}{ll} \arg\max\limits_{\mathbf{P}}\ \inf\limits_{\mathbf{w}} & \{D_{\mathbf{w}}^\Gamma(\mathbf{P})\}\ , \\ s.t. & P_i \geq P_{il},\ i=0,\ldots,L-1 \end{array}$$

where $D_{\mathbf{w}}^\Gamma = R_{\mathbf{w}}^\Gamma - R_{basis}^\Gamma$. When uncertainty is global, this hyperplane is defined by the risk in the L extreme cases with $P_i = \delta_{ik}$, that is, by the corresponding c_{ii}. However, with partial knowledge of the prior probabilities, this hyperplane becomes defined by the risk in L points which are the vertex given by the restrictions and with associated risk denoted by c_{jj}^Γ.

Defining

$$l(P_i) = \frac{1}{1+\exp^{-\tau(P_i-P_{il})}}\ , \tag{26}$$

where τ controls the hardness of this restriction, the minimax problem can be re-formulated as

$$\begin{array}{ll} \arg\max\limits_{\mathbf{P}}\ \inf\limits_{\mathbf{w}} & \{D_{\mathbf{w}}^\Gamma(\mathbf{P})\} \\ s.t. & l(P_i) \geq 1/2,\ i=0,\ldots,L-1. \end{array}$$

Thus, this constrained optimization problem can be solved as a non-constrained problem by considering an auxiliary function that incorporates the restriction as a barrier function

$$\arg\max_{\mathbf{P}} \inf_{\mathbf{w}} \{D^{\Gamma}_{\mathbf{w}}(\mathbf{P}) + A\psi(\mathbf{P})\} \ ,$$

where $\psi(P_i) = \log(l(P_i))$ and the constant A determines the contribution of the barrier function.

Fig. 4(b) shows the new risk function corresponding to the binary case previously depicted in Fig. 4(a). Note that, it is the sum of the original $R_B(\mathbf{P})$ and the barrier function $\psi(\mathbf{P})$.

As in Section 4.1, in order to derive the network weight learning rule, we need to compute

$$\begin{aligned}
\frac{\partial\widehat{\psi}}{\partial B_i} &= \sum_{j=0}^{L-1} \frac{\partial\widehat{\psi}}{\partial\widehat{P}_j}\frac{\partial\widehat{P}_j}{\partial B_i}, \\
&= \tau\widehat{P}_i\sum_{k=0}^{L-1}\left(1-l(\widehat{P}_k)\right)(\delta_{ik}-\widehat{P}_k), \\
&= \tau\widehat{P}_i\widehat{\psi}_{di} \ ,
\end{aligned}$$

where $\widehat{\psi}_{di} = \sum_{k=0}^{L-1}(1-l(\widehat{P}_k))(\delta_{ik}-\widehat{P}_k)$

As τ increases, the constraints become harder around the specified bound.

The update learning rule for the auxiliary variable B_i at cycle n is

$$B_i^{(n+1)} = B_i{}^{(n)} + \rho\widehat{P}_i^{(n)}\widehat{R}_{di}^{\Gamma(n)} + \rho A\tau\widehat{P}_i^{(n)}\widehat{\psi}_{di}^{(n)} \ .$$

And therefore, using (23), the update learning rule for P_i is

$$\widehat{P}_i^{(n+1)} = \frac{\widehat{P}_i^{(n)}\exp\left(\rho\widehat{P}_i^{(n)}\widehat{R}_{di}^{\Gamma(n)}\right)\exp\left(\rho A\tau\widehat{P}_i^{(n)}\widehat{\psi}_{di}^{(n)}\right)}{\sum_{j=0}^{L-1}\left\{\widehat{P}_j^{(n)}\exp\left(\rho\widehat{P}_j^{(n)}\widehat{R}_{dj}^{\Gamma(n)}\right)\exp\left(\rho A\tau\widehat{P}_j^{(n)}\widehat{\psi}_{dj}^{(n)}\right)\right\}} \ .$$

Note that if the upper bound is known instead of the lower bound, $l(P_i)$ defined by (26) should be replaced by $u(P_i) = (1+\exp(\tau(P_i - P_{iu})))^{-1}$ at the previous formulation.

The minimax constrained optimization problem has been tackled by considering a new objective function defined by the sum of the original cost function and a barrier function. Studying the convexity of this new function becomes important from the fact that a stationary point of this risk curve is a global maximum.

Since the minimum risk curve ($R_B(P)$) is a convex function of the priors (see VanTrees, 1968, for details), if we verify the convexity of the barrier function, we can conclude that the function defined by the sum of both of them is also convex.

This barrier function is convex in $\mathcal{P}$ if the Hessian matrix $\mathbf{H}_R$ verifies $\mathbf{P}^{\mathbf{T}}\mathbf{H}_R\mathbf{P} \le 0$

The Hessian matrix of the barrier function equals to a diagonal matrix $\mathbf{D}_{\mathbf{r}} = diag(\mathbf{r})$ with all negative diagonal entries $r_i = A\tau^2(-l(P_i)(1-l(P_i)))$. As $l(P_i) \in [0,1]$ and therefore, $r_i \le 0$, it is straightforward to see that

$$\begin{aligned}
\mathbf{P}^{\mathbf{T}}\mathbf{H}_R\mathbf{P} &= \mathbf{P}^{\mathbf{T}}\mathbf{D}_{\mathbf{r}}\mathbf{P}, \\
&= \sum_{i=0}^{L-1} P_i^2 r_i \le 0 \ .
\end{aligned}$$

Since the barrier function is convex, the new objective function (defined by the sum of two convex functions) is also convex.

5.2 Extension to Other Learning Algorithms

The learning algorithm proposed in this paper is intended to train a minimax deviation classifier based on neural networks with feedforward architecture. Actually, the learning algorithm we propose becomes a feasible solution for any learning process based on minimizing some empirical estimate of an overall cost (error) function.

However, it is also applicable to a general classifier provided it is trained (in an iterative process) for the estimated minimax deviation probabilities and the assumed decision costs. Specifically, in this paper, scaling the learning rate allows to simulate different class distributions and the hard decisions are made based on posterior probability estimates and decision costs. Furthermore, the neural learning phase carried out in one iteration can be re-used for the next one, what allows to reduce computational cost with respect to a complete optimization process on each iteration. Apart from the general approach of completely training a classifier on each iteration and in order to reduce its computational cost, specific solutions may be studied for different learning machines. Nonetheless, it seems not feasible to readily achieve this improvement for classifiers like SVMs, where support vectors for one solution may have nothing in common with the ones obtained in next iteration and thus, making necessary to re-train the classifier in each iteration.

Another possible solution for any classifier that provides a posteriori probabilities estimates or any score that can be converted into probabilities (for details on calibration methods see Wei et al., 1999; Zadrozny and Elkan, 2002; Niculescu-Mizil and Caruana, 2005) is outlined here. In this case, an iterative procedure able to estimate the minimax deviation probabilities and consequently to adjust (without re-training) the outputs of the classifier could be studied. The general idea for this approach is as follows: first, the new minimax deviation prior probabilities are estimated according to (25) and then, posterior probabilities provided by the model are adjusted as follows (see Saerens et al., 2002, for more details)

$$P^{(k)}\{\mathbf{d}=\mathbf{u}_i|\mathbf{x}\} = \frac{\dfrac{P_i^{(k)}}{P_i^{(k-1)}} P^{(k-1)}\{\mathbf{d}=\mathbf{u}_i|\mathbf{x}\}}{\sum_{j=0}^{L-1} \dfrac{P_j^{(k)}}{P_j^{(k-1)}} P^{(k-1)}\{\mathbf{d}=\mathbf{u}_j|\mathbf{x}\}} \ . \tag{27}$$

The algorithm's main structure is summarized as

```
for $k = 1$ to $K$ do
    Estimate $\widehat{R}^{(k)}, \widehat{R}_i^{(k)}, i = 0, \ldots, L-1$, according to (21), (22) and decision costs $c_{ij}$
    Update minimax probability $\widehat{P}_i^{(k+1)}$ according to (25)
    Adjust classifier outputs according to (27)
end for
```

The effectiveness of this method relies on the accuracy of the initial *a posteriori* probability estimates. Studying in depth this approach and comparing different minimax deviation classifiers (decision trees, SVMs, RBF networks, feedforward networks and committee machines) together with different probability calibration methods appears as a challenging issue to be explored in future work.

6. Experimental Results

In this section, we first present the neural network architecture used in the experiments and illustrate the proposed *minimax deviation* strategy on an artificial data set. Then, we apply it to several real-world classification problems. Moreover, a comparison with other proposals such as the traditional *minimax* and the common *re-balancing* approach is carried out.

6.1 Softmax-based Network

Although our algorithms can be applied to any classifier architecture, we have chosen a neural network based on the softmax non-linearity with soft decisions given by

$$y_i = \sum_{j=1}^{M_i} y_{ij} \ ,$$

with

$$y_{ij} = \frac{\exp(\mathbf{w}_{ij}^T \mathbf{x} + w_{ij0})}{\sum_{k=0}^{L-1} \sum_{l=1}^{M_k} \exp(\mathbf{w}_{kl}^T \mathbf{x} + w_{kl0})} \ ,$$

where L stands for the number of classes, M_j the number of softmax outputs used to compute y_j and $\mathbf{w}_{ij}$ are weight vectors. We will refer to this network as a *Generalized Softmax Perceptron*(GSP).[1] A simple network with $M_j = 2$ is used in the experiments.

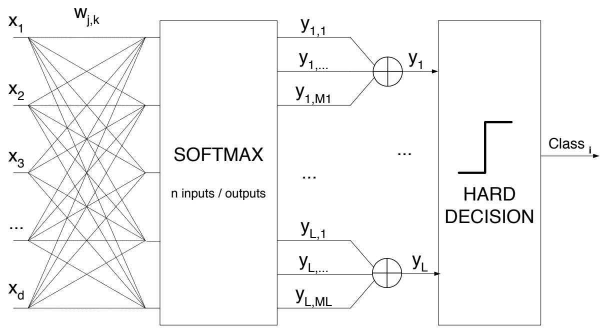


Figure 5: GSP(Generalized Softmax Perceptron) Network

Fig. 5 corresponds to the neural network architecture used to classify the samples represented by feature vector $\mathbf{x}$. Learning consists of estimating network parameters $\mathbf{w}$ by means of the stochastic gradient minimization of certain objective functions. In the experiments, we have considered the *Cross Entropy* objective function given by

$$CE(\mathbf{y}, \mathbf{d}) = -\sum_{i=1}^{L} d_i \log y_i \ .$$

The stochastic gradient learning rule for the GSP network is given by Eq. (18). Learning step $\mu^{(k)}$ decreases according to $\mu^{(k)} = \frac{\mu^{(0)}}{1+k/\eta}$, where k is the iteration number, $\mu^{(0)}$ the initial learning rate and η a decay factor.

1. Note that the GSP is similar to a two layer MLP with a single layer of weights and with coupled saturation function (softmax), instead of sigmoidal units.

The reason to illustrate this approach with a feedforward architecture is that, as mentioned in Section 5.2, it allows to exploit (in the iterative learning process) the partially optimized solution in current iteration for the next one. On the other hand, posterior probability estimation makes it possible to apply the adaptive strategy based on prior re-estimation proposed by Saerens to the minimax deviation classifier, as long as a data set representative of the operation conditions is available. Finally, the fact that intermediate outputs y_{ij} of the GSP can be interpreted as subclass probabilities may provide quite a natural way to cope with the unexplored problem of uncertainty in subclass distributions as already pointed out by Webb and Ting (2005). Nonetheless, both architecture and cost function issues are not the goal of this paper, but merely illustrative tools.

6.2 Artificial Data Set

To illustrate the *minimax regret* approach proposed in this paper both under complete and partial uncertainty, an artificial data set with two classes (class $\mathbf{u}_0$ and class $\mathbf{u}_1$) has been created. Data examples are drawn from the normal distribution $p(\mathbf{x}|\mathbf{d}=\mathbf{u}_i)=N(m_i,\sigma_i^2)$ with mean m_i and standard deviation σ_i. Mean values were set to $m_0=0, m_1=2$ and standard deviation to $\sigma_0=\sigma_1=\sqrt{2}$. A total of 4000 instances were generated with prior probabilities of class membership $P\{\mathbf{d}=\mathbf{u}_0\}=0.93$ and $P\{\mathbf{d}=\mathbf{u}_1\}=0.07$. The cost-benefit matrix $\begin{pmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{pmatrix}$ is given by $\begin{pmatrix} 2 & 5 \\ 4 & 0 \end{pmatrix}$.

Initial learning rate was set to $\mu^{(0)}=0.3$, decay factor to $\eta=2000$ and training was ended after 80 cycles. Classifier assessment was carried out by following 10-fold cross-validation.

Two classifiers were trained, to be called a *standard classifier* and a *minMaxDev classifier*. The former is built by considering that the estimated class prior information is precise and stationary and the latter is the approach proposed in this paper to cope with uncertainty in priors. Thus, for the *standard classifier*, its performance may deviate from the optimal risk in 3.39 when priors change from training to test conditions. However, a *minimax deviation classifier* reduces this worst-case difference from the optimal classifier to 0.77.

Now, we suppose that some information about priors is available (partial uncertainty). For instance, we consider that the lower bound for prior probabilities P_0 and P_1 are known and set to $P_{0l}=0.55$ and $P_{1l}=0.05$, respectively, so that the uncertainty region is $\Gamma=\{(P_0,P_1)|P_0\in[0.55,0.95], P_1\in[0.05,0.45]\}$.

A minimax deviation classifier can be derived for Γ (it will be called Γ-*minMaxDev classifier*).The narrower Γ is, the closer the minimax deviation classifier performance is to the optimal. For this particular case, under partially imprecise priors, the *standard classifier* may differ from optimal (in Γ) in 0.83, while the use of the simple *minMaxDev classifier* designed under total prior uncertainty conditions attains a maximum deviation of 0.53. However, the Γ-*minMaxDev classifier* only differs from optimal in 0.24. These data are reported in Table 1 where both, experimental and also theoretical results, are shown.

6.3 Real Databases

In this section we report experimental results obtained with several publicly available data sets. From the UCI repository (Blake and Merz, 1998) the following benchmarks: German Credits, Australian Credits, Insurance Company, DNA slice-junction, Page-blocks, Dermatology and Pen-digits.

	Classifier		
	Standard Th/Exp	minMaxDev Th/Exp	Γ-minMaxDev Th/Exp
Maximum deviation from optimal (complete uncertainty)	3.41/*3.39*	**0.72/*0.77***	–
Maximum deviation from optimal in Γ (partial uncertainty)	0.85/*0.83*	0.50/*0.53*	**0.19/*0.24***

Table 1: A comparison between the *standard* classifier (build under stationary prior assumptions), the minimax deviation classifier (*minMaxDev*) and the minimax deviation classifier under partial uncertainty (Γ-*minMaxDev*) for an artificial data set

Database	**# Classes**	**Class distribution**	**# Attributes**	**# Instances**
German Credits (**GCRE**)	2	[0.70 0.30]	8	1000
Australian Credits (**AUS**)	2	[0.32 0.68]	14	690
Munich Credits (**MCRE**)	2	[0.30 0.70]	20	1000
Insurance Company (**COIL**)	2	[0.94 0.06]	85	9822
DNA Slice-junction (**DNA**)	3	[0.24 0.24 0.52]	180	3186
Page-blocks (**PAG**)	5	[0.90 0.06 0.01 0.01 0.02]	10	5473
Dermatology (**DER**)	6	[0.31 0.16 0.20 0.13 0.14 0.06]	34	366
Pen-digits (**PEN**)	10	[0.104 0.104 0.104 0.096 0.104 0.096 0.096 0.104 0.096 0.096]	16	10992

Table 2: Experimental Data sets

Other public data set used is Munich Credits from the Dept. of Statistics at the University of Munich.[2]

Data set description is summarized in Table 2, and cost-benefit matrices are shown in Table 3. We have used the cost values that appear in Ikizler (2002) for those data sets in common. Otherwise, for lack of an expert analyst, the cost values have been chosen by hand.

2. Data sets available at *http://www.stat.uni-muenchen.de/service/datenarchiv/welcome_e.html*.

Insurance Company

$$\begin{pmatrix} 0 & 0 \\ 1 & -17 \end{pmatrix}$$

German, Australian, Munich Credits

$$\begin{pmatrix} -1 & 5 \\ 0 & 0 \end{pmatrix}$$

DNA

$$\begin{pmatrix} -1 & 2 & 3 \\ 2 & -1 & 3 \\ 2 & 2 & 0 \end{pmatrix}$$

Page-Blocks

$$\begin{pmatrix} -1 & 1 & 1 & 1 & 1 \\ 2 & 0 & 1 & 1 & 1 \\ 2 & 1 & 0 & 1 & 1 \\ 2 & 1 & 1 & 0 & 1 \\ 2 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Dermatology

$$\begin{pmatrix} -4 & 3 & 3 & 2 & 2 & 2 \\ 2 & -3 & 3 & 2 & 1 & 3 \\ 3 & 3 & -8 & 4 & 4 & 5 \\ 4 & 5 & 5 & -10 & 5 & 2 \\ 3 & 1 & 4 & 3 & -6 & 3 \\ 4 & 5 & 5 & 4 & 5 & -10 \end{pmatrix}$$

Pendigits

$$c_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{Otherwise} \end{cases}$$

Table 3: Cost-Benefit matrices for the experimental Data sets

	Maximum Risk Deviation from the optimal classifier			
	Standard	Re-balanced	Minimax Deviation *minMaxDev*	Minimax *minMax*
GCRE	0.70	0.80	(0.55 **0.60**)	0.99
ACRE	1.00	1.00	(0.76 **0.86**)	1.00
MCRE	0.91	0.77	(0.54 **0.59**)	0.99
COIL	2.78	0.99	(0.87 **0.92**)	16.32
DNA	0.34	0.53	(**0.30** 0.27 0.25)	1.14
PAG	0.62	0.26	(0.13 0.13 **0.20** 0.16 0.16)	0.86
DER	1.03	1.28	(0.67 **0.78** 0.51 0.48 0.54 0.60)	7.62
PEN	0.061	0.059	(0.024 0.025 0.023 0.026 0.023 0.028 0.019 0.021 0.022 **0.029**)	**0.029**

Table 4: Classifier Performance evaluated as Maximum Risk Deviation from the optimal classifier for several real-world applications. Class-conditional risk deviations ($R_i - c_{ii}$) reported for the *minMaxDev* classifier.

Experimental results for these data sets are shown in the following sections. The robustness of different decision machines under complete uncertainty of prior probabilities is analyzed in Section 6.3.1. If uncertainty is only partial, a similar study and comparison with the previous approach (complete uncertainty) is carried out in Section 6.3.2.

6.3.1 CLASSIFIER ROBUSTNESS UNDER COMPLETE UNCERTAINTY

We now study how different neural-based classifiers cope with worst-case situations in prior probabilities. The maximum deviation from the optimal classifier (see Table 4) is reported for the proposed *minMaxDev* strategy as well as for other alternative approaches: the one based on the assumption of stationary priors (*standard*) and the common alternative of deriving the classifier from an equally distributed data set (*re-balanced*). A comparison with the traditional minimax strategy is also provided. Together with the previously mentioned value (maximum deviation or regret), deviation for the L class-conditional extreme cases ($R_i - c_{ii}$) is also reported for the *minMaxDev* classifier in Table 4. Results allow to verify that this solution is fairly close to the optimal one where deviation is not dependent on priors and thus, class-conditional deviations take the same value.

Although the balanced class distribution to train the classifier can be obtained by means of undersampling and/or oversampling, it is simulated by altering the learning rate used in the training phase according to (19) as $\mu_i = \mu \frac{1/L}{\widehat{P}_i^{(0)}}$, where $1/L$ represents the simulated probability, equal for all classes.

Results evidence that the assumption of stationary priors may lead to significant deviations from the optimal decision rule under "unexpected", but rather realistic, prior changes. This deviation may reach up to three times more than the robust minimax deviation strategy. Thus, for classification problems like Page-blocks the maximum deviation from the optimal classifier is **0.62** for the

	Maximum Risk			
	Standard	Re-balanced	Minimax Deviation	Minimax
			minMaxDev	*minMax*
GCRE	0.70	0.15	0.60	**0.00**
ACRE	0.01	0.02	0.86	**-0.00**
MCRE	0.05	0.20	0.59	**0.00**
COIL	0.76	0.99	0.86	**0.02**
DNA	0.34	0.53	0.25	**0.13**
PAG	0.62	0.26	0.20	**0.10**
DER	-2.10	-1.68	-2.21	**-2.38**
PEN	0.061	0.059	**0.029**	**0.029**

Table 5: Classifier Performance measured as Maximum Risk for several real-world applications.

standard classifier while this reduces to **0.20** for the *minMaxDev* one. Likewise, for the Insurance company(COIL) application the maximum deviation for the *standard classifier* is **2.78** compared with **0.92** for the *minMaxDev* model. The remaining databases also show the same behavior as it is presented in Table 4.

On the other hand, the use of a classifier inferred from a re-balanced data set does not necessarily involve a decrease in the maximum deviation with respect to the *standard classifier*. In the same way, the traditional minimax classifier does not protect against prior changes in terms of maximum relative deviation from the minimum risk classifier.

However, if our criterion is more conservative and our aim is the minimization of the maximum possible risk (not the minimization of the deviation), the traditional minimax classifier represents the best option. It is shown in Table 5 where the maximum risk for the different classifiers is reported. Positive values in this table indicate a cost while negative values represent a benefit. For instance, for the Page-blocks application the minimax classifier assures a maximum risk of **0.10** while the *standard*, *re-balanced* and *minMaxDev* classifiers reach values of 0.62, 0.26 and 0.20, respectively. It can be noticed that for the Pen-digits data set, the minimax deviation and minimax approaches attain the same results. The reason is that, for this problem, the R_{basis} plane takes the same value (in this case, zero) in the probability space.

6.3.2 CLASSIFIER ROBUSTNESS UNDER PARTIAL UNCERTAINTY

Unlike the previous section, we consider now that partial information about the class priors is available. The aim is to find a classifier that behaves well for a delimited and realistic range of priors what constitutes an aid in reducing the maximum deviation from the optimal classifier. This situation can be treated as a constrained minimax regret strategy where the constraints represent any extra information about prior probability value.

Experimental results for several situations of partial prior uncertainty are presented in this section. We consider that lower bounds for the prior probabilities are available (see Table 6). In order to get the Γ-*minMaxDev classifier*, the risk for the different vertex of the uncertainty domain needs to be calculated. With them, the basis risk R^{Γ}_{basis} over which deviations are measured is derived.

	Lower bound for prior probabilities									
Data Set	P_{0l}	P_{1l}	P_{2l}	P_{3l}	P_{4l}	P_{5l}	P_{6l}	P_{7l}	P_{8l}	P_{9l}
GCRE	0.40	0.25								
ACRE	0.20	0.25								
MCRE	0.20	0.25								
COIL	0.15	0.03								
DNA	0.10	0.10	0.25							
PAG	0.22	0.02	0.00	0.01	0.02					
DER	0.1	0.20	0.10	0.10	0.10	0.02				
PEN	0.10	0.06	0.06	0.10	0.10	0.06	0.06	0.10	0.05	0.05

Table 6: Lower bounds for prior probabilities defining the uncertainty region, Γ region for the experimental data sets.

	Maximum Risk Deviation in the uncertainty region		
	Standard	Minimax Deviation *minMaxDev*	Minimax Deviation with restriction Γ-*minMaxDev*
GCRE	0.24	0.19	(**0.10** 0.09)
ACRE	**0.03**	0.64	(**0.03** **0.03**)
MCRE	0.22	0.38	(**0.13** 0.10)
COIL	2.33	0.77	(**0.17** 0.11)
DNA	0.14	0.08	(**0.07** **0.07** 0.06)
PAG	0.37	0.15	(**0.10** 0.08 0.08 0.05 0.04)
DER	0.08	**0.05**	(0.03 0.03 0.04 0.02 **0.05** **0.05**)
PEN	0.013	0.007	(**0.003** **0.003** 0.001 0.000 0.001 0.001 0.000 0.001 **0.003** 0.001)

Table 7: Classifier Performance under partial knowledge of prior probabilities measured as Maximum Risk Deviation for several real-world applications. Class-conditional risk deviations ($R_i^\Gamma - c_{ii}^\Gamma$) are reported for the Γ-*minMaxDev* classifier.

Maximum deviation from the optimal in Γ is reported for the Γ-*minMaxDev* classifier together with the *standard* and the *minMaxDev* ones. For instance, the *standard classifier* for the Page-blocks data set deviates from the optimal classifier, in the defined uncertainty region, up to 0.37, while when complete uncertainty is assumed the maximum deviation is equal to 0.62.

In the same way, reducing the uncertainty also means a reduction in the maximum deviation for *minMaxDev* classifier (trained without considering this partial knowledge). Thus, for Γ, this classifier assures a deviation bound of 0.15. However, taking into account this partial information to train a Γ-*minMaxDev* classifier allows to reduce the deviation for the worst-case conditions to 0.10. It can be seen the same behavior for the other databases in Table 7.

7. Conclusions

This work concerns the design of robust neural-based classifiers when the prior probabilities of the classes are partially or completely unknown, even by the end user.

This problem of uncertainty in the class priors is often ignored in supervised classification, even though it is a widespread situation in real world applications. As a result, the reliability of the inducted classifier can be greatly affected as previously shown by the experiments.

To tackle this problem, we have proposed a novel *minimax deviation* strategy with the goal to minimize the maximum deviation with respect to the optimal classifier.

A neural network training algorithm based on learning rate scaling has been developed. The experimental results show that this minimax deviation (*minMaxDev*) classifier protects against prior changes while other approaches like ignoring this uncertainty or use a balanced learning data set may result in large differences in performance with respect to the minimum risk classifier. Also, it has been shown that the conventional minimax classifier reduces the maximum possible risk following a conservative attitude but at the expense of large worst-case differences from the optimal classifier.

Furthermore, a constrained minimax deviation approach (Γ-*minMaxDev*) has been derived for those situations where uncertainty is only partial. This may be seen as a general approach with some particular cases: a) precise knowledge of prior probabilities and b) complete uncertainty about the priors. In a) the region of uncertainty collapses to a point and we have the Bayes' rule of minimum risk and in b) the pure minimax deviation strategy comes up. While the first one may be criticized for being quite unrealistic, the other may be seen rather pessimistic. The experimental results for this proposed intermediate situation show that the Γ-*minMaxDev* classifier allows to reduce the maximum deviation from the optimal and performs well over a range of prior probabilities.

Acknowledgments

The authors thank the four referees and the associate editor for their helpful comments. This work was partially supported by the project TEC2005-06766-C03-02 from the Spanish Ministry of Education and Science.

References

N. Abe, B. Zadrozny, and J. Langford. An iterative method for multi-class cost-sensitive learning. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 3–11, 2004.

N. M. Adams and D. J. Hand. Comparing classifiers when the misallocation costs are uncertain. *Pattern Recognition*, 32(7):1139–1147, March 1998.

R. Alaiz-Rodriguez, A. Guerrero-Curieses, and J. Cid-Sueiro. Minimax classifiers based on neural networks. *Pattern Recognition*, 38(1):29–39, January 2005.

R. Barandela, J. S. Sanchez, V. García, and E. Rangel. Strategies for learning in class imbalance problems. *Pattern Recognition*, 36(3):849–851, March 2003.

J. O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer, second edition, 1985.

C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998. URL `http://www.ics.uci.edu/ mlearn/MLRepository.html`.

L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Chapman & Hall, NY, 1984.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

J. Cid-Sueiro and A. R. Figueiras-Vidal. On the structure of strict sense Bayesian cost functions and its applications. *IEEE Transactions on Neural Networks*, 12(3):445–455, May 2001.

C. Drummond and R. C. Holte. Explicitly representing expected cost: An alternative to ROC representation. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 198–207. ACM Press, 2000.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley and Sons, 2001.

Y. C. Eldar and N. Merhav. Minimax approach to robust estimation of random parameters. *IEEE Trans. on Signal Processing*, 52(7):1931–1946, July 2004.

Y. C. Eldar, A. Ben-Tal, and A. Nemirovski. Linear minimax regret estimation of deterministic parameters with bounded data uncertainties. *IEEE Trans. on Signal Processing*, 52(8):2177–2188, August 2004.

M. Feder and N. Merhav. Universal composite hypothesis testing: A competitive minimax approach. *IEEE Trans. on Information Theory*, 48(6):1504–1517, June 2002.

A. Guerrero-Curieses, R. Alaiz-Rodriguez, and J. Cid-Sueiro. A fixed-point algorithm to minimax learning with neural networks. *IEEE Transactions on Systems, Man and Cybernetics Part C*, 34 (4):383–392, November 2004.

H. A. Güvenir, N. Emeksiz, N. Ikizler, and N. Örmeci. Diagnosis of gastric carcinoma by classification on feature projections. *Artificial Intelligence in Medicine*, 31(3), 2004.

N. Ikizler. Benefit maximizing classification using feature intervals. Technical Report BU-CE-0208, Bilkent University, Ankara, Turkey, 2002.

N. Japkowicz and S. Stephen. The class imbalance problem: A systematic study. *Intelligent Data Analysis Journal*, 6(5):429–450, November 2002.

M. G. Kelly, D. J. Hand, and N. M. Adams. The impact of changing populations on classifier performance. In *Proceedings of Fifth International Conference on SIG Knowledge Discovery and Data Mining (SIGKDD)*, pages 367–371, San Diego, CA, 1999.

H. J. Kim. On a constrained optimal rule for classification with unknown prior individual group membership. *Journal of Multivariate Analysis*, 59(2):166–186, November 1996.

M. Kubat and S. Matwin. Addressing the curse of imbalanced training sets: One-sided selection. In *Proceedings 14th International Conference on Machine Learning*, pages 179–186. Morgan Kaufmann, 1997.

M. Kubat, R. Holte, and S. Matwin. Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, 30(2/3):195–215, 1998.

S. Lawrence, I. Burns, A. D. Back, A. C. Tsoi, and C. L. Giles. Neural network classification and unequal prior class probabilities. In G. Orr, K.-R. Müller, and R. Caruana, editors, *Tricks of the Trade*, Lecture Notes in Computer Science State-of-the-Art Surveys, pages 299–314. Springer Verlag, 1998.

T. K. Moon and W. C. Stirling. *Mathematical Methods and Algorithms for Signal Processing*. Prentice Hall, 2000.

A. Niculescu-Mizil and R. Caruana. Predicting good probabilities with supervised learning. In *ICML '05: Proceedings of the 22nd International Conference on Machine learning*, pages 625–632, New York, NY, USA, 2005. ACM Press. ISBN 1-59593-180-5.

E. Polak. *Optimization: Algorithms and Consistent Approximations*. Springer, 1997.

F. Provost. Learning with imbalanced data sets 101. In *Invited paper for the AAAI 2000 Workshop on Imbalanced Data Sets*. AAAI Press. Technical Report WS-00-05, 2000.

F. Provost and T. Fawcett. Robust classification systems for imprecise environments. *Machine Learning*, 42(3):203–231, March 2001.

M. Saerens, P. Latinne, and C. Decaestecker. Adjusting a classifier for new a priori probabilities: A simple procedure. *Neural Computation*, 14:21–41, January 2002.

E. Takimoto and M. Warmuth. The minimax strategy for Gaussian density estimation. In *Proceedings 13th Annual Conference on Computational Learning Theory*, pages 100–106. Morgan Kaufmann, San Francisco, 2000.

K. M. Ting. A study of the effect of class distribution using cost-sensitive learning. In *Proceedings of the Fifth International Conference on Discovery Science*, pages 98–112. Berlin: Springer-Verlag, 2002.

H. L. VanTrees. *Detection, Estimation and Modulation Theory*. John Wiley and Sons, 1968.

G. I. Webb and K. M. Ting. On the application of ROC analysis to predict classification performance under varying class distributions. *Machine Learning*, 58(1):25–32, 2005.

W. Wei, T. K. Leen, and E. Barnard. A fast histogram-based postprocessor that improves posterior probability estimates. *Neural Computation*, 11(5):1235 – 1248, July 1999.

B. Zadrozny and C. Elkan. Learning and making decisions when costs and probabilities are both unknown. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 204–213. ACM Press, 2001.

B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Eighth International Conference on Knowledge Discovery and Data Mining*, 2002.

B. Zadrozny, J. Langford, and N. Abe. Cost-sensitive learning by cost-proportionate example weighting. In *Proceedings of the third IEEE International Conference on Data Mining*, pages 435–442, 2003.

Z. H. Zhou and X. Y. LiuJ. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):63–77, January 2006.

Journal of Machine Learning Research 8 (2007) 131-154 Submitted 2/05; Revised 9/05; Published 1/07

Distances between Data Sets Based on Summary Statistics

Nikolaj Tatti NTATTI@CC.HUT.FI
HIIT Basic Research Unit
Laboratory of Computer and Information Science
Helsinki University of Technology, Finland

Editor: Dale Schuurmans

Abstract

The concepts of similarity and distance are crucial in data mining. We consider the problem of defining the distance between two data sets by comparing summary statistics computed from the data sets. The initial definition of our distance is based on geometrical notions of certain sets of distributions. We show that this distance can be computed in cubic time and that it has several intuitive properties. We also show that this distance is the unique Mahalanobis distance satisfying certain assumptions. We also demonstrate that if we are dealing with binary data sets, then the distance can be represented naturally by certain parity functions, and that it can be evaluated in linear time. Our empirical tests with real world data show that the distance works well.

Keywords: data mining theory, complex data, binary data, itemsets

1. Introduction

In this paper we will consider the following problem: Given two data sets D_1 and D_2 of dimension K, define a distance between D_1 and D_2. To be more precise, we consider the problem of defining the distance between two multisets of transactions, each set sampled from its own unknown distribution. We will define a dissimilarity measure between D_1 and D_2 and we will refer to this measure as *CM distance*.

Generally speaking, the notion of dissimilarity between two objects is one of the most fundamental concepts in data mining. If one is able to retrieve a distance matrix from a set of objects, then one is able to analyse data by using for example, clustering or visualisation techniques. Many real world data collections may be naturally divided into several data sets. For example, if a data collection consists of movies from different eras, then we may divide the movies into subcollections based on their release years. A distance between these data (sub)sets would provide means to analyse them as single objects. Such an approach may ease the task of understanding complex data collections.

Let us continue by considering the properties the CM distance should have. First of all, it should be a metric. The motivation behind this requirement is that the metric theory is a well-known area and metrics have many theoretical and practical virtues. Secondly, in our scenario the data sets have statistical nature and the CM distance should take this into account. For example, consider that both data sets are generated from the same distribution, then the CM distance should give small values and approach 0 as the number of data points in the data sets increases. The third requirement is that we should be able to evaluate the CM distance quickly. This requirement is crucial since we may have high dimensional data sets with a vast amount of data points.

The CM distance will be based on summary statistics, features. Let us give a simple example: Assume that we have data sets $D_1 = \{A,B,A,A\}$ and $D_2 = \{A,B,C,B\}$ and assume that the only feature we are interested in is the proportion of A in the data sets. Then we can suggest the distance between D_1 and D_2 to be $|3/4 - 1/4| = 1/2$. The CM distance is based on this idea; however, there is a subtle difficulty: If we calculate several features, then should we take into account the correlation of these features? We will do exactly that in defining the CM distance.

The rest of this paper is organised as follows. In Section 2 we give the definition of the CM distance by using some geometrical interpretations. We also study the properties of the distance and provide an alternative characterisation. In Section 3 we study the CM distance and binary data sets. In Section 4 we discuss how the CM distance can be used with event sequences and in Section 5 we comment about the feature selection. Section 6 is devoted for related work. The empirical tests are represented in Section 7 and we conclude our work with the discussion in Section 8.

2. The Constrained Minimum Distance

In the following subsection we will define our distance using geometrical intuition and show that the distance can be evaluated efficiently. In the second subsection we will discuss various properties of the distance, and in the last subsection we will provide an alternative justification to the distance. The aim of this justification is to provide more theoretical evidence for our distance.

2.1 The Definition

We begin by giving some basic definitions. By a *data set D* we mean a finite collection of samples lying in some finite space Ω. The set Ω is called *sample space*, and from now on we will denote this space by the letter Ω. The number of elements in Ω is denoted by $|\Omega|$. The number of samples in the data set D is denoted by $|D|$.

As we said in the introduction, our goal is not to define a distance directly on data sets but rather through some statistics evaluated from the data sets. In order to do so, we define a *feature function* $S : \Omega \to \mathbb{R}^N$ to map a point in the sample space to a real vector. Throughout this section S will indicate some given feature function and N will indicate the dimension of the range space of S. We will also denote the i^{th} component of S by S_i. Note that if we have several feature functions, then we can join them into one big feature function. A *frequency* $\theta \in \mathbb{R}^N$ of S taken with respect to a data set D is the average of values of S taken over the data set, that is, $\theta = \frac{1}{|D|}\sum_{\omega \in D} S(\omega)$. We denote this frequency by $S(D)$.

Although we do not make any assumptions concerning the size of Ω, some of our choices are motivated by thinking that $|\Omega|$ can be very large—so large that even the simplest operation, say, enumerating all the elements in Ω, is not tractable. On the other hand, we assume that N is such that an algorithm executable in, say, $O(N^3)$ time is feasible. In other words, we seek a distance whose evaluation time does not depend of the size of Ω but rather of N.

Let $\mathbb{P}$ be the set of all distributions defined on Ω. Given a feature function S and a frequency θ (calculated from some data set) we say that a distribution $p \in \mathbb{P}$ satisfies the frequency θ if $\mathrm{E}_p[S] = \theta$. We also define a *constrained set of distributions*

$$\mathcal{C}_+(S,\theta) = \{p \in \mathbb{P} \mid \mathrm{E}_p[S] = \theta\}$$

to be the set of the distributions satisfying θ. The idea behind this is as follows: From a given data set we calculate some statistics, and then we examine the distributions that can produce such frequencies.

We interpret the sets $\mathbb{P}$ and $\mathcal{C}_+(S,\theta)$ as *geometrical objects*. This is done by enumerating the points in Ω, that is, we think that $\Omega = \{1,2,,\ldots,|\Omega|\}$. We can now represent each distribution $p \in \mathbb{P}$ by a vector $u \in \mathbb{R}^{|\Omega|}$ by setting $u_i = p(i)$. Clearly, $\mathbb{P}$ can be represented by the vectors in $\mathbb{R}^{|\Omega|}$ having only non-negative elements and summing to one. In fact, $\mathbb{P}$ is a simplex in $\mathbb{R}^{|\Omega|}$. Similarly, we can give an alternative definition for $\mathcal{C}_+(S,\theta)$ by saying

$$\mathcal{C}_+(S,\theta) = \left\{ u \in \mathbb{R}^{|\Omega|} \mid \sum_{i\in\Omega} S(i)u_i = \theta, \sum_{i\in\Omega} u_i = 1, u \geq 0 \right\}. \tag{1}$$

Let us now study the set $\mathcal{C}_+(S,\theta)$. In order to do so, we define a *constrained space*

$$\mathcal{C}(S,\theta) = \left\{ u \in \mathbb{R}^{|\Omega|} \mid \sum_{i\in\Omega} S(i)u_i = \theta, \sum_{i\in\Omega} u_i = 1 \right\},$$

that is, we drop the last condition from Eq. 1. The set $\mathcal{C}_+(S,\theta)$ is included in $\mathcal{C}(S,\theta)$; the set $\mathcal{C}_+(S,\theta)$ consists of the non-negative vectors from $\mathcal{C}(S,\theta)$. Note that the constraints defining $\mathcal{C}(S,\theta)$ are vector products. This implies that $\mathcal{C}(S,\theta)$ is an affine space, and that, given two different frequencies θ_1 and θ_2, the spaces $\mathcal{C}(S,\theta_1)$ and $\mathcal{C}(S,\theta_2)$ are parallel.

Example 1 *Let us illustrate the discussion above with a simple example. Assume that $\Omega = \{A,B,C\}$. We can then imagine the distributions as vectors in $\mathbb{R}^3$. The set $\mathbb{P}$ is the triangle having $(1,0,0)$, $(0,1,0)$, and $(0,0,1)$ as corner points (see Figure 1). Define a feature function S to be*

$$S(\omega) = \begin{cases} 1 & \omega = C \\ 0 & \omega \neq C. \end{cases}$$

The frequency $S(D)$ is the proportion of C in a data set D. Let $D_1 = (C,C,C,A)$ and $D_2 = (C,A,B,A)$. Then $S(D_1) = 0.75$ and $S(D_2) = 0.25$. The spaces $\mathcal{C}(S,0.25)$ and $\mathcal{C}(S,0.75)$ are parallel lines (see Figure 1). The distribution sets $\mathcal{C}_+(S,0.25)$ and $\mathcal{C}_+(S,0.75)$ are the segments of the lines $\mathcal{C}(S,0.25)$ and $\mathcal{C}(S,0.75)$, respectively.

The idea of interpreting distributions as geometrical objects is not new. For example, a well-known boolean query problem is solved by applying linear programming to the constrained sets of distributions (Hailperin, 1965; Calders, 2003).

Let us revise some elementary Euclidean geometry: Assume that we are given two parallel affine spaces $\mathcal{A}_1$ and $\mathcal{A}_2$. There is a natural way of measuring the distance between these two spaces. This is done by taking the length of the shortest segment going from a point in $\mathcal{A}_1$ to a point in $\mathcal{A}_2$ (for example see the illustration in Figure 1). We know that the segment has the shortest length if and only if it is orthogonal with the affine spaces. We also know that if we select a point $a_1 \in \mathcal{A}_1$ having the shortest norm, and if we similarly select $a_2 \in \mathcal{A}_2$, then the segment going from a_1 to a_2 has the shortest length.

The preceding discussion and the fact that the constrained spaces are affine motivates us to give the following definition: Assume that we are given two data sets, namely D_1 and D_2 and a

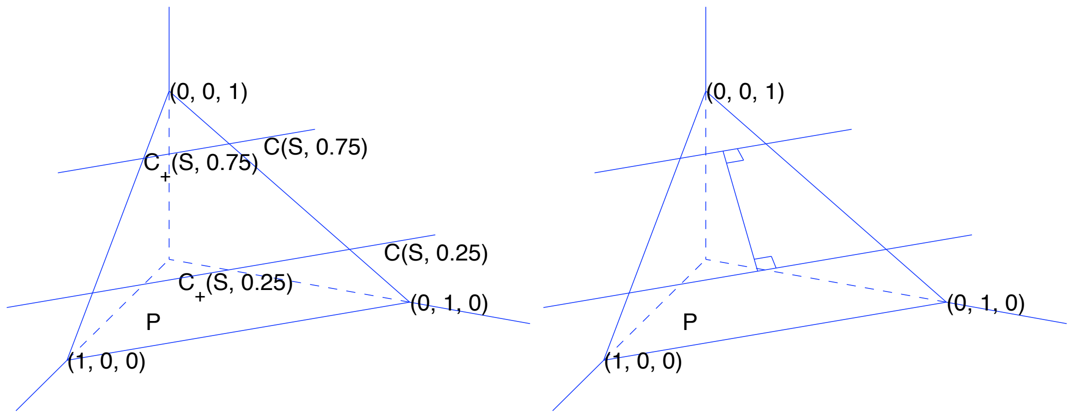


Figure 1: A geometrical interpretation of the distribution sets for $|\Omega| = 3$. In the left figure, the set $\mathbb{P}$, that is, the set of all distributions, is a triangle. The constrained spaces $\mathcal{C}(S, 0.25)$ and $\mathcal{C}(S, 0.75)$ are parallel lines and the distribution sets $\mathcal{C}_+(S, 0.25)$ and $\mathcal{C}_+(S, 0.75)$ are segments of the constrained spaces. In the right figure we added a segment perpendicular to the constraint spaces. This segment has the shortest length among the segments connecting the constrained spaces.

feature function S. Let us shorten the notation $\mathcal{C}(S, S(D_i))$ by $\mathcal{C}(S, D_i)$. We pick a vector from each constrained space having the shortest norm

$$u_i = \operatorname*{argmin}_{u \in \mathcal{C}(S, D_i)} \|u\|_2, \quad i = 1, 2.$$

We define the distance between D_1 and D_2 to be

$$d_{CM}(D_1, D_2 \mid S) = \sqrt{|\Omega|}\, \|u_1 - u_2\|_2. \tag{2}$$

The reasons for having the factor $\sqrt{|\Omega|}$ will be given later. We will refer to this distance as *Constrained Minimum (CM) distance*. We should emphasise that u_1 or u_2 may have negative elements. Thus the CM distance is *not* a distance between two distributions; it is rather a distance based on the frequencies of a given feature function and is motivated by the geometrical interpretation of the distribution sets.

The main reason why we define the CM distance using the constrained spaces $\mathcal{C}(S, D_i)$ and not the distribution sets $\mathcal{C}_+(S, D_i)$ is that we can evaluate the CM distance efficiently. We discussed earlier that Ω may be very large so it is crucial that the evaluation time of a distance does not depend on $|\Omega|$. The following theorem says that the CM distance can be represented using the frequencies and a covariance matrix

$$\operatorname{Cov}[S] = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} S(\omega) S(\omega)^T - \left(\frac{1}{|\Omega|} \sum_{\omega \in \Omega} S(\omega) \right) \left(\frac{1}{|\Omega|} \sum_{\omega \in \Omega} S(\omega) \right)^T.$$

Theorem 1 *Assume that* Cov $[S]$ *is invertible. For the CM distance between two data sets D_1 and D_2 we have*

$$d_{CM}(D_1, D_2 \mid S)^2 = (\theta_1 - \theta_2)^T \operatorname{Cov}^{-1}[S] (\theta_1 - \theta_2),$$

where $\theta_i = S(D_i)$.

The proofs for the theorems are given in Appendix.

The preceding theorem shows that we can evaluate the distance using the covariance matrix and frequencies. If we assume that evaluating a single component of the feature function S is a unit operation, then the frequencies can be calculated in $O(N|D_1| + N|D_2|)$ time. The evaluation time of the covariance matrix is $O(|\Omega| N^2)$ but we assume that S is such that we know a closed form for the covariance matrix (such cases will be discussed in Section 3), that is, we assume that we can evaluate the covariance matrix in $O(N^2)$ time. Inverting the matrix takes $O(N^3)$ time and evaluating the distance itself is $O(N^2)$ operation. Note that calculating frequencies and inverting the covariance matrix needs to be done only once: for example, assume that we have k data sets, then calculating the distances between every data set pair can be done in $O\left(N \sum_i^k |D_i| + N^3 + k^2 N^2\right)$ time.

Example 2 *Let us evaluate the distance between the data sets given in Example 1 using both the definition of the CM distance and Theorem 1. We see that the shortest vector in $\mathcal{C}(S, 0.25)$ is $u_1 = \left(\frac{3}{8}, \frac{3}{8}, \frac{1}{4}\right)$. Similarly, the shortest vector in $\mathcal{C}(S, 0.75)$ is $u_2 = \left(\frac{1}{8}, \frac{1}{8}, \frac{3}{4}\right)$. Thus the CM distance is equal to*

$$d_{CM}(D_1, D_2 \mid S) = \sqrt{3}\, \|u_1 - u_2\|_2 = \sqrt{3} \left[\frac{2^2}{8^2} + \frac{2^2}{8^2} + \frac{2^2}{4^2}\right]^{1/2} = \frac{3}{\sqrt{8}}.$$

The covariance of S is equal to $\operatorname{Cov}[S] = \frac{1}{3} - \frac{1}{3}\frac{1}{3} = \frac{2}{9}$. *Thus Theorem 1 gives us*

$$d_{CM}(D_1, D_2 \mid S) = \left[\operatorname{Cov}^{-1}[S] \left(\frac{3}{4} - \frac{1}{4}\right)^2\right]^{1/2} = \left[\frac{9}{2}\left(\frac{2}{4}\right)^2\right]^{1/2} = \frac{3}{\sqrt{8}}.$$

From Theorem 1 we see a reason to have the factor $\sqrt{|\Omega|}$ in Eq. 2: Assume that we have two data sets D_1 and D_2 and a feature function S. We define a new sample space $\Omega' = \{(\omega, b) \mid \omega \in \Omega, b = 0, 1\}$ and transform the original data sets into new ones by setting $D_i' = \{(\omega, 0) \mid \omega \in D_i\}$. We also expand S into Ω' by setting $S'(\omega, 1) = S'(\omega, 0) = S(\omega)$. Note that $S(D_i) = S'(D_i')$ and that $\operatorname{Cov}[S] = \operatorname{Cov}[S']$ so Theorem 1 says that the CM distance has not changed during this transformation. This is very reasonable since we did not actually change anything essential: We simply added a bogus variable into the sample space, and we ignored this variable during the feature extraction. The size of the new sample space is $|\Omega'| = 2|\Omega|$. This means that the difference $\|u_1 - u_2\|_2$ in Eq. 2 is smaller by the factor $\sqrt{2}$. The factor $\sqrt{|\Omega|}$ is needed to negate this effect.

2.2 Properties

We will now list some important properties of $d_{CM}(D_1, D_2 \mid S)$.

Theorem 2 *$d_{CM}(D_1, D_2 \mid S)$ is a pseudo metric.*

The following theorem says that adding external data set to the original data sets makes the distance smaller which is very reasonable property.

Theorem 3 *Assume three data sets D_1, D_2, and D_3 over the same set of items. Assume further that D_1 and D_2 have the same number of data points and let $\varepsilon = \frac{|D_3|}{|D_1| + |D_3|}$. Then*

$$d_{CM}(D_1 \cup D_3, D_2 \cup D_3 \mid S) = (1 - \varepsilon) d_{CM}(D_1, D_2 \mid S).$$

Theorem 4 *Let A be a $M \times N$ matrix and b a vector of length M. Define $T(\omega) = AS(\omega) + b$. It follows that $d_{CM}(D_1, D_2 \mid T) \leq d_{CM}(D_1, D_2 \mid S)$ for any D_1 and D_2.*

Corollary 5 *Adding extra feature functions cannot decrease the distance.*

Corollary 6 *Let A be an invertible $N \times N$ matrix and b a vector of length N. Define $T(\omega) = AS(\omega) + b$. It follows that $d_{CM}(D_1, D_2 \mid T) = d_{CM}(D_1, D_2 \mid S)$ for any D_1 and D_2.*

Corollary 6 has an interesting interpretation. Note that $T(D) = AS(D) + b$ and that $S(D) = A^{-1}(T(D) - b)$. This means that if we know the frequencies $S(D)$, then we can infer the frequencies $T(D)$ without a new data scan. Similarly, we can infer $S(D)$ from $T(D)$. We can interpret this relation by thinking that $S(D)$ and $T(D)$ are merely different representations of the same feature information. Corollary 6 says that the CM distance is equal for any such representation.

2.3 Alternative Characterisation of the CM Distance

We derived our distance using geometrical interpretation of the distribution sets. In this section we will provide an alternative way for deriving the CM distance. Namely, we will show that if some distance is of Mahalanobis type and satisfies some mild assumptions, then this distance is proportional to the CM distance. The purpose of this theorem is to provide more theoretical evidence to our distance.

We say that a distance d is of Mahalanobis type if

$$d(D_1, D_2 \mid S)^2 = (\theta_1 - \theta_2)^T C(S)^{-1} (\theta_1 - \theta_2),$$

where $\theta_1 = S(D_1)$ and $\theta_2 = S(D_2)$ and $C(S)$ maps a feature function S to a symmetric $N \times N$ matrix. Note that if $C(S) = \operatorname{Cov}[S]$, then the distance d is the CM distance. We set $\mathbb{M}$ to be the collection of all distances of Mahalanobis type. Can we justify the decision that we examine only the distances included in $\mathbb{M}$? One reason is that a distance belonging to $\mathbb{M}$ is guaranteed to be a metric. The most important reason, however, is the fact that we can evaluate the distance belonging to $\mathbb{M}$ efficiently (assuming, of course, that we can evaluate $C(S)$).

Let $d \in \mathbb{M}$ and assume that it satisfies two additional assumptions:

1. If A is an $M \times N$ matrix and b is a vector of length M and if we set $T(\omega) = AS(\omega) + b$, then $C(T) = AC(S)A^T$.
2. Fix two points ω_1 and ω_2. Let $\sigma : \Omega \to \Omega$ be a function swapping ω_1 and ω_2 and mapping everything else to itself. Define $U(\omega) = S(\sigma(\omega))$. Then $d(\sigma(D_1), \sigma(D_2) \mid U) = d(D_1, D_2 \mid S)$.

The first assumption can be partially justified if we consider that A is an invertible square matrix. In this case the assumption is identical to $d(\cdot, \cdot \mid AS + b) = d(\cdot, \cdot \mid S)$. This is to say that the distance is independent of the representation of the frequency information. This is similar to Corollary 6 given in Section 2.2. We can construct a distance that would satisfy Assumption 1 in the invertible case but fail in a general case. We consider such distances pathological and exclude them by making a broader assumption. To justify Assumption 2 note that the frequencies have not changed, that is, $U(\sigma(D)) = S(D)$. Only the representation of single data points have changed. Our argument is that the distance should be based on the frequencies and not on the values of the data points.

Theorem 7 *Let $d \in M$ satisfying Assumptions 1 and 2. If $C(S)$ is invertible, then there is a constant $c > 0$, not depending on S, such that $d(\cdot, \cdot \mid S) = c d_{CM}(\cdot, \cdot \mid S)$.*

3. The CM distance and Binary Data Sets

In this section we will concentrate on the distances between binary data sets. We will consider the CM distance based on itemset frequencies, a very popular statistics in the literature concerning binary data mining. In the first subsection we will show that a more natural way of representing the CM distance is to use parity frequencies. We also show that we can evaluate the distance in linear time. In the second subsection we will provide more theoretical evidence why the CM distance is a good distance between binary data sets.

3.1 The CM Distance and Itemsets

We begin this section by giving some definitions. We set the sample space Ω to be

$$\Omega = \{\omega \mid \omega = (\omega_1, \ldots, \omega_K), \omega_i = 0, 1\},$$

that is, Ω is the set of all binary vectors of length K. Note that $|\Omega| = 2^K$. It is custom that each dimension in Ω is identified with some symbol. We do this by assigning the symbol a_i to the i^{th} dimension. These symbols are called *attributes* or *items*. Thus when we speak of the attribute a_i we refer to the i^{th} dimension. We denote the set of all items by $\mathbb{A} = \{a_1, \ldots, a_K\}$. A non-empty subset of $\mathbb{A}$ is called *itemset*.

A *boolean formula* $S : \Omega \to \{0, 1\}$ is a feature function mapping a binary vector to a binary value. We are interested in two particular boolean formulae: Assume that we are given an itemset $B = \{a_{i_1}, \ldots, a_{i_L}\}$. We define a *conjunction function* S_B to be

$$S_B(\omega) = \omega_{i_1} \land \omega_{i_2} \land \cdots \land \omega_{i_K},$$

that is, S_B results 1 if and only if all the variables corresponding the itemset B are on. Given a data set D the frequency $S_B(D)$ is called the frequency of the itemset B. Conjunction functions are popular and there are a lot of studies in the literature concerning finding itemsets that have large frequency (see e.g., Agrawal et al., 1993; Hand et al., 2001). We also define a *parity function* T_B to be

$$T_B(\omega) = \omega_{i_1} \oplus \omega_{i_2} \oplus \cdots \oplus \omega_{i_K},$$

where $\oplus$ is the binary operator XOR. The function T_B results 1 if and only if the number of active variables included in B are odd.

A collection $\mathcal{F}$ of itemsets is said to be *antimonotonic* or *downwardly closed* if each non-empty subset of an itemset included in $\mathcal{F}$ is also included in $\mathcal{F}$. Given a collection of itemsets $\mathcal{F} = \{B_1, \ldots, B_N\}$ we extend our definition of the conjunction function by setting $S_{\mathcal{F}} = [S_{B_1}, \ldots, S_{B_N}]^T$. We also define $T_{\mathcal{F}} = [T_{B_1}, \ldots, T_{B_N}]^T$.

Assume that we are given an antimonotonic family $\mathcal{F}$ of itemsets. We can show that there is an invertible matrix A such that $T_{\mathcal{F}} = AS_{\mathcal{F}}$. In other words, we can get the parity function $T_{\mathcal{F}}$ from the conjunction function $T_{\mathcal{F}}$ by an invertible linear transformation. Corollary 6 now implies that

$$d_{CM}(D_1, D_2 \mid S_{\mathcal{F}}) = d_{CM}(D_1, D_2 \mid T_{\mathcal{F}}), \tag{3}$$

for any D_1 and D_2. The following lemma shows that the covariance matrix $\operatorname{Cov}[T_{\mathcal{F}}]$ of the parity function is very simple.

Lemma 8 *Let $T_{\mathcal{F}}$ be a parity function for a family of itemsets $\mathcal{F}$, then* $\operatorname{Cov}[T_{\mathcal{F}}] = 0.5I$*, that is, the covariance matrix is a diagonal matrix having* 0.5 *at the diagonal.*

Theorem 1, Lemma 8, and Eq. 3 imply that

$$d_{CM}(D_1, D_2 \mid S_{\mathcal{F}}) = \sqrt{2}\left\|\theta_1 - \theta_2\right\|_2, \tag{4}$$

where $\theta_1 = T_{\mathcal{F}}(D_1)$ and $\theta_2 = T_{\mathcal{F}}(D_2)$. This identity says that the CM distance can be calculated in $O(N)$ time (assuming that we know the frequencies θ_1 and θ_2). This is better than $O(N^3)$ time implied by Theorem 1.

Example 3 *Let $I = \left\{\{a_j\} \mid j = 1 \ldots K\right\}$ be a family of itemsets having only one item. Note that $T_{\{a_j\}} = S_{\{a_j\}}$. Eq. 4 implies that*

$$d_{CM}(D_1, D_2 \mid S_I) = \sqrt{2}\left\|\theta_1 - \theta_2\right\|_2,$$

where θ_1 and θ_2 consists of the marginal frequencies of each a_j calculated from D_1 and D_2, respectively. In this case the CM distance is simply the L_2 distance between the marginal frequencies of the individual attributes. The frequencies θ_1 and θ_2 resemble term frequencies (TF) used in text mining (see e.g., Baldi et al., 2003).

Example 4 *We consider now a case with a larger set of features. Our motivation for this is that using only the feature functions S_I is sometimes inadequate. For example, consider data sets with two items having the same individual frequencies but different correlations. In this case the data sets may look very different but according to our distance they are equal.*

Let $C = I \cup \left\{a_j a_k \mid j, k = 1 \ldots K, j < k\right\}$ be a family of itemsets such that each set contains at most two items. The corresponding frequencies contain the individual means and the pairwise correlation for all items. Let $S_{a_j a_k}$ be the conjunction function for the itemset $a_j a_k$. Let $\gamma_{jk} = S_{a_j a_k}(D_1) - S_{a_j a_k}(D_2)$ be the difference between the correlation frequencies. Also, let $\gamma_j = S_{a_j}(D_1) - S_{a_j}(D_2)$. Since

$$T_{a_j a_k} = S_{a_j} + S_{a_k} - 2S_{a_j a_k}$$

it follows from Eq. 4 that

$$d_{CM}(D_1, D_2 \mid S_C)^2 = 2\sum_{j<k}\left(\gamma_j + \gamma_k - 2\gamma_{jk}\right)^2 + 2\sum_{j=1}^{K}\gamma_j^2. \tag{5}$$

3.2 Characterisation of the CM Distance for Itemsets

The identity given in Eq. 4 is somewhat surprising and seems less intuitive. A question arises: why this distance is more natural than some other, say, a simple L_2 distance between the itemset frequencies. Certainly, parity functions are less intuitive than conjunction functions. One answer is that the parity frequencies are decorrelated version of the traditional itemset frequencies.

However, we can clarify this situation from another point of view: Let $\mathcal{A}$ be the set of all itemsets. Assume that we are given two data sets D_1 and D_2 and define *empirical distributions* p_1 and p_2 by setting

$$p_i(\omega) = \frac{\text{number of samples in } D_i \text{ equal to } \omega}{|D_i|}.$$

The constrained spaces of $S_{\mathcal{A}}$ are singular points containing only p_i, that is, $\mathcal{C}(S_{\mathcal{A}}, D_i) = \{p_i\}$. This implies that

$$d_{CM}(D_1, D_2 \mid S_{\mathcal{A}}) = \sqrt{2^K} \left\| p_1 - p_2 \right\|_2 . \tag{6}$$

In other words, the CM distance is proportional to the L_2 distance between the empirical distributions. This identity seems very reasonable. At least, it is more natural than, say, taking L_2 distance between the traditional itemset frequencies.

The identity in Eq. 6 holds only when we use the features $S_{\mathcal{A}}$. However, we can prove that a distance of the Mahalanobis type satisfying the identity in Eq. 6 and some additional conditions is equal to the CM distance. Let us explain this in more detail. We assume that we have a distance d having the form

$$d(D_1, D_2 \mid S_{\mathcal{F}})^2 = (\theta_1 - \theta_2)^T C(S_{\mathcal{F}})^{-1} (\theta_1 - \theta_2),$$

where $\theta_1 = S_{\mathcal{F}}(D_1)$ and $\theta_2 = S_{\mathcal{F}}(D_2)$ and $C(S_{\mathcal{F}})$ maps a conjunction function $S_{\mathcal{F}}$ to a symmetric $N \times N$ matrix. The distance d should satisfy the following mild assumptions.

1. Assume two antimonotonic families of itemsets $\mathcal{F}$ and $\mathcal{H}$ such that $\mathcal{F} \subset \mathcal{H}$. It follows that $d(\cdot, \cdot \mid S_{\mathcal{F}}) \leq d(\cdot, \cdot \mid S_{\mathcal{H}})$.
2. Adding extra dimensions (but not changing the features) does not change the distance.

The following theorem says that the assumptions and the identity in Eq. 6 are sufficient to prove that d is actually the CM distance.

Theorem 9 *Assume that a Mahalanobis distance d satisfies Assumptions 1 and 2. Assume also that there is a constant c_1 such that*

$$d(D_1, D_2 \mid S_{\mathcal{A}}) = c_1 \left\| p_1 - p_2 \right\|_2 .$$

Then it follows that for any antimonotonic family $\mathcal{F}$ we have

$$d(D_1, D_2 \mid S_{\mathcal{F}}) = c_2 d_{CM}(D_1, D_2 \mid S_{\mathcal{F}}),$$

for some constant c_2.

4. The CM distance and Event Sequences

In the previous section we discussed about the CM distance between the binary data sets. We will use similar approach to define the CM distance between sequences.

An *event sequence s* is a finite sequence whose symbols belong to a finite alphabet Σ. We denote the length of the event sequence s by $|s|$, and by $s(i, j)$ we mean a subsequence starting from i and ending at j. The subsequence $s(i, j)$ is also known as *window*. A popular choice for statistics of event sequences are *episodes* (Hand et al., 2001). A *parallel episode* is represented by a subset of the alphabet Σ. A window of s satisfies a parallel episode if all the symbols given in the episode occur in the window. Assume that we are given an integer k. Let W be a collection of windows of s having the length k. A *frequency* of a parallel episode is the proportion of windows in W satisfying the episode. We should point out that this mapping destroys the exact ordering of the sequence. On the other hand, if some symbols occur often close to each other, then the episode consisting of these symbols will have a high frequency.

In order to apply the CM distance we will now describe how we can transform a sequence s to a binary data set. Assume that we are given a window length k. We transform a window of length k into a binary vector of length $|\Sigma|$ by setting 1 if the corresponding symbol occurs in the window, and 0 otherwise. Let D be the collection of these binary vectors. We have now transformed the sequence s to the binary data set D. Note that parallel episodes of s are represented by itemsets of D.

This transformation enables us to use the CM distance. Assume that we are given two sequences s_1 and s_2, a collection of parallel episodes $\mathcal{F}$, and a window length k. First, we transform the sequences into data sets D_1 and D_2. We set the CM distance between the sequences s_1 and s_2 to be $d_{CM}(D_1, D_2 \mid S_{\mathcal{F}})$.

5. Feature Selection

We will now discuss briefly about feature selection—a subject that we have taken for granted so far. The CM distance depends on a feature function S. How can we choose a good set of features?

Assume for simplicity that we are dealing with binary data sets. Eq. 6 tells us that if we use all itemsets, then the CM distance is L_2 distance between empirical distributions. However, to get a reliable empirical distribution we need an exponential number of data points. Hence we can use only some subset of itemsets as features. The first approach is to make an expert choice without seeing data. For example, we could decide that the feature function is S_I, the means of the individual attributes, or S_C, the means of individual attributes and the pairwise correlation.

The other approach is to infer a feature function from the data sets. At first glimpse this seems an application of feature selection. However, traditional feature selection fails: Let S_I be the feature function representing the means of the individual attributes and let $S_{\mathcal{A}}$ be the feature function containing all itemsets. Let ω be a binary vector. Note that if we know $S_I(\omega)$, then we can deduce $S_{\mathcal{A}}(\omega)$. This means that S_I is a *Markov blanket* (Pearl, 1988) for $S_{\mathcal{A}}$. Hence we cannot use the Markov blanket approach to select features. The essential part is that the traditional feature selection algorithms deal with the *individual* points. We try to select features for whole data sets.

Note that feature selection algorithms for singular points are based on training data, that is, we have data points divided into clusters. In other words, when we are making traditional feature selection we *know* which points are close and which are not. In order to make the same ideas work for data sets we need to have similar information, that is, we need to know which data sets are close to each other, and which are not. Such an information is rarely provided and hence we are forced to seek some other approach.

We suggest a simple approach for selecting itemsets by assuming that frequently occurring itemsets are interesting. Assume that we are given a collection of data sets D_i and a threshold σ. Let I be the itemsets of order one. We define $\mathcal{F}$ such that $B \in \mathcal{F}$ if and only if $B \in I$ or that B is a σ-frequent itemset for some D_i.

6. Related Work

In this section we discuss some existing methods for comparing data sets and compare the evaluation algorithms. The execution times are summarised in Table 1.

Distance	Time
CM distance (general case)	$O(NM+N^2\|\Omega\|+N^3)$
CM distance (known cov. matrix)	$O(NM+N^3)$
CM distance (binary case)	$O(NM+N)$
Set distances	$O(M^3)$
Kullback-Leibler	$O(NM+N\|\Omega\|)$
Fischer's Information	$O(NM+N^2\|D_2\|+N^3)$

Table 1: Comparison of the execution times of various distances. The number $M = |D_1| + |D_2|$ is the number of data points in total. The $O(NM)$ term refers to the time needed to evaluate the frequencies $S(D_1)$ and $S(D_2)$. Kullback-Leibler distance is solved using Iterative Scaling algorithm in which one round has N steps and one step is executed in $O(|\Omega|)$ time.

6.1 Set Distances

One approach to define a data set distance is to use some natural distance between single data points and apply some known set distance. Eiter and Mannila (1997) show that some data set distances defined in this way can be evaluated in cubic time. However, this is too slow for us since we may have a vast amount of data points. The other downsides are that these distances may not take into account the statistical nature of data which may lead into problems.

6.2 Edit Distances

We discuss in Section 4 of using the CM distance for event sequences. Traditionally, edit distances are used for comparing event sequences. The most famous edit distance is Levenshtein distance (Levenshtein, 1966). However, edit distances do not take into account the statistical nature of data. For example, assume that we have two sequences generated such that the events are sampled from the uniform distribution independently of the previous event (a zero-order Markov chain). In this case the CM distance is close to 0 whereas the edit distance may be large. Roughly put, the CM distance measures the dissimilarity between the statistical characteristics whereas the edit distances operate at the symbol level.

6.3 Minimum Discrimination Approach

There are many distances for distributions (see Baseville, 1989, for a nice review). From these distances the CM distance resembles the statistical tests involved with Minimum Discrimination Theorem (see Kullback, 1968; Csiszár, 1975). In this framework we are given a feature function S and two data sets D_1 and D_2. From the set of distributions $\mathcal{C}_+(S, D_i)$ we select a distribution maximising the entropy and denote it by p_i^{ME}. The distance itself is the Kullback-Leibler divergence between p_1^{ME} and p_2^{ME}. It has been empirically shown that p_i^{ME} represents well the distribution from which D_i is generated (see Mannila et al., 1999). The downsides are that this distance is not a metric (it is not even symmetric), and that the evaluation time of the distance is infeasible: Solving p_i^{ME} is **NP**-hard (Cooper, 1990). We can approximate the Kullback-Leibler distance by Fischer's information, that is,

$$D\left(p_1^{ME} \| p_2^{ME}\right) \approx \frac{1}{2}\left(\theta_1 - \theta_2\right)^T \operatorname{Cov}^{-1}\left[S \mid p_2^{ME}\right]\left(\theta_1 - \theta_2\right),$$

where $\theta_i = S(D_i)$ and $\operatorname{Cov}\left[S \mid p_2^{ME}\right]$ is the covariance matrix of S taken with respect to p_2^{ME} (see Kullback, 1968). This resembles greatly the equation in Theorem 1. However, in this case the covariance matrix depends on data sets and thus generally this approximation is not a metric. In addition, we do not know p_2^{ME} and hence we cannot evaluate the covariance matrix. We can, however, estimate the covariance matrix from D_2, that is,

$$\operatorname{Cov}\left[S \mid p_2^{ME}\right] \approx \frac{1}{|D_2|}\sum_{\omega \in D_2} S(\omega)S(\omega)^T - \frac{1}{|D_2|^2}\left[\sum_{\omega \in D_2} S(\omega)\right]\left[\sum_{\omega \in D_2} S(\omega)^T\right].$$

The execution time of this operation is $O(N^2 |D_2|)$.

7. Empirical Tests

In this section we describe our experiments with the CM distance. We begin by examining the effect of different feature functions. We continue studying the distance by applying clustering algorithms, and finally we represent some interpretations to the results.

In many experiments we use a base distance d_U defined as the L_2 distance between the itemset frequencies, that is,

$$d_U\left(D_1, D_2 \mid S\right) = \sqrt{2}\left\|\theta_1 - \theta_2\right\|_2, \tag{7}$$

where θ_i are the itemset frequencies $\theta_i = S(D_i)$. This type of distance was used in Hollmén et al. (2003). Note that $d_U\left(D_1, D_2 \mid ind\right) = d_{CM}\left(D_1, D_2 \mid ind\right)$, where ind is the feature set containing only individual means.

7.1 Real World Data Sets

We examined the CM distance with several real world data sets and several feature sets. We had 7 data sets: *Bible*, a collection of 73 books from the Bible,[1] *Addresses*, a collection of 55 inaugural addresses given by the presidents of the U.S.,[2] *Beatles*, a set of lyrics from 13 studio albums made by the Beatles, *20Newsgroups*, a collection of 20 newsgroups,[3] *TopGenres*, plot summaries for top rated movies of 8 different genres, and *TopDecades*, plot summaries for top rated movies from 8 different decades.[4] *20Newsgroups* contained (in that order) 3 religion groups, 3 of politics, 5 of computers, 4 of science, 4 recreational, and *misc.forsale*. *TopGenres* consisted (in that order) of *Action*, *Adventure*, *SciFi*, *Drama*, *Crime*, *Horror*, *Comedy*, and *Romance*. The decades for *TopDecades* were 1930–2000. Our final data set, *Abstract*, was composed of abstracts describing NSF awards from 1990–1999.[5]

1. The books were taken from `http://www.gutenberg.org/etext/8300` on July 20, 2005.
2. The addresses were taken from `http://www.bartleby.com/124/` on August 17, 2005.
3. The data set was taken from `http://www.ai.mit.edu/~jrennie/20Newsgroups/`, a site hosted by Jason Rennie, on June 8, 2001.
4. The movie data sets were taken from `http://www.imdb.com/Top/` on January 1, 2006.
5. The data set was taken from `http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.data.html` on January 13, 2006.

Bible and *Addresses* were converted into binary data sets by taking subwindows of length 6 (see the discussion in Section 4). We reduced the number of attributes to 1000 by using the mutual information gain. *Beatles* was preprocessed differently: We transformed each song to its binary bag-of-words representation and selected 100 most informative words. In *20Newsgroups* a transaction was a binary bag-of-words representation of a single article. Similarly, In *TopGenres* and in *TopDecades* a transaction corresponded to a single plot summary. We reduced the number of attributes in these three data sets to 200 by using the mutual information gain. In *Abstract* a data set represented one year and a transaction was a bag-of-words representation of a single abstract. We reduced the dimension of *Abstract* to 1000.

7.2 The Effect of Different Feature Functions

We begin our experiments by studying how the CM distance (and the base distance) changes as we change features.

We used 3 different sets of features: *ind*, the independent means, *cov*, the independent means along with the pairwise correlation, and *freq*, a family of frequent itemsets obtained by using APRIORI (Agrawal et al., 1996). We adjusted the threshold so that *freq* contained $10K$ itemsets, where K is the number of attributes.

We plotted the CM distances and the base distances as functions of $d_{CM}(ind)$. The results are shown in Figure 2. Since the number of constraints varies, we normalised the distances by dividing them with $\sqrt{N}$, where N is the number of constraints. In addition, we computed the correlation of each pair of distances. These correlations are shown in Table 2.

	d_{CM} vs. d_{CM}			d_U vs. d_U			d_{CM} vs. d_U	
	cov	*freq*	*freq*	*cov*	*freq*	*freq*	*cov*	*freq*
Data set	*ind*	*ind*	*cov*	*ind*	*ind*	*cov*	*cov*	*freq*
20Newsgroups	0.996	0.725	0.733	0.902	0.760	0.941	0.874	0.571
Addresses	1.000	0.897	0.897	0.974	0.927	0.982	0.974	0.743
Bible	1.000	0.895	0.895	0.978	0.946	0.989	0.978	0.802
Beatles	0.982	0.764	0.780	0.951	0.858	0.855	0.920	0.827
TopGenres	0.996	0.817	0.833	0.916	0.776	0.934	0.927	0.931
TopDecades	0.998	0.735	0.744	0.897	0.551	0.682	0.895	0.346
Abstract	1.000	0.985	0.985	0.996	0.993	0.995	0.996	0.994
Total	0.998	0.702	0.709	0.934	0.894	0.938	0.910	0.607

Table 2: Correlations for various pairs of distances. A column represents a pair of distances and a row represents a single data set. For example, the correlation between $d_{CM}(ind)$ and $d_{CM}(cov)$ in *20Newsgroups* is 0.996. The last row is the correlation obtained by using the distances from all data sets simultaneously. Scatterplots for the columns 1–2 and 4–5 are given in Fig. 2.

Our first observation from the results is that $d_{CM}(cov)$ resembles $d_{CM}(ind)$ whereas $d_{CM}(freq)$ produces somewhat different results.

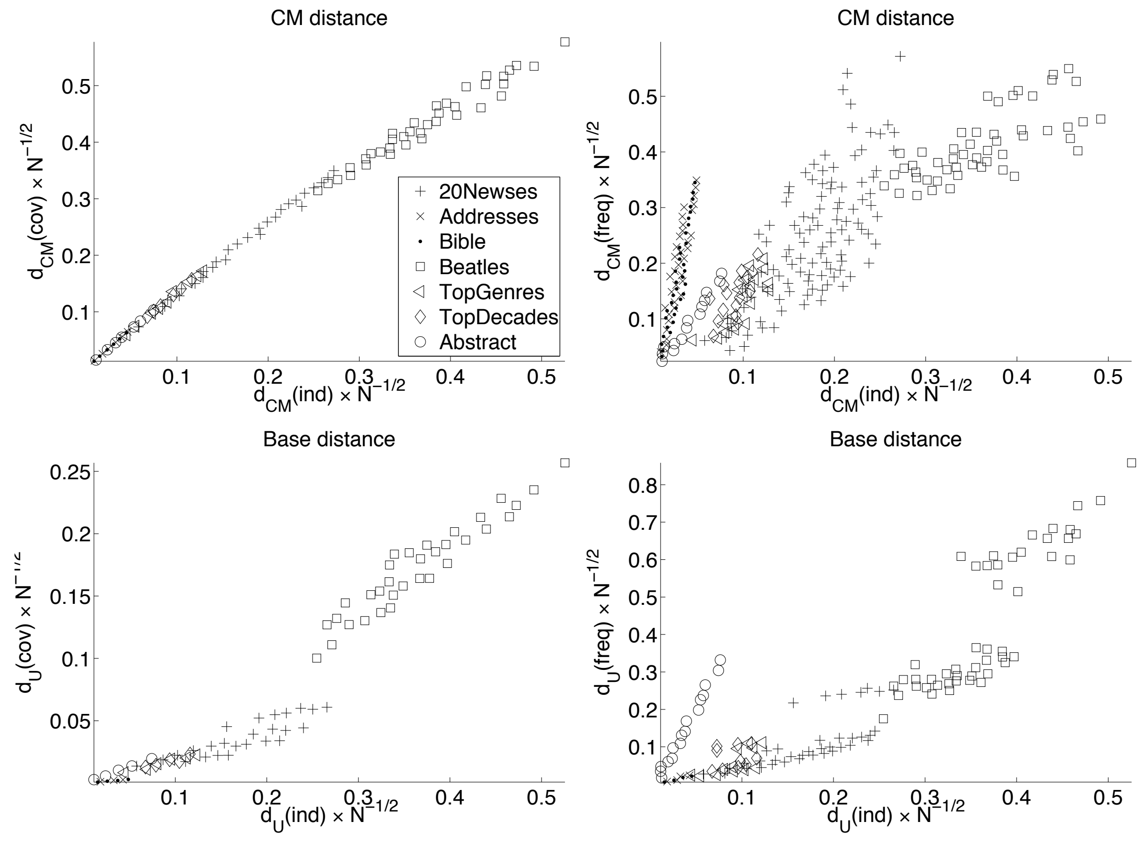


Figure 2: CM and base distances as functions of $d_{CM}(ind)$. A point represents a distance between two data sets. The upper two figures contain the CM distances while the lower two contain the base distance. The distances were normalised by dividing $\sqrt{N}$, where N is the number of constraints. The corresponding correlations are given in Table 2. Note that x-axis in the left (right) two figures are equivalent.

The correlations between $d_{CM}(cov)$ and $d_{CM}(ind)$ are stronger than the correlations between $d_U(cov)$ and $d_U(ind)$. This can be explained by examining Eq. 5 in Example 4. If the dimension is K, then the itemsets of size 1, according to Eq. 5, involve $\frac{1}{2}K(K-1)+K$ times in computing $d_{CM}(cov)$, whereas in computing $d_U(cov)$ they involve only K times. Hence, the itemsets of size 2 have smaller impact in $d_{CM}(cov)$ than in $d_U(cov)$.

On the other hand, the correlations between $d_{CM}(freq)$ and $d_{CM}(ind)$ are weaker than the correlations between $d_U(freq)$ and $d_U(ind)$, implying that the itemsets of higher order have stronger impact on the CM distance.

7.3 Clustering Experiments

In this section we continue our experiments by applying clustering algorithms to the distances. Our goal is to compare the clusterings obtained from the CM distance to those obtained from the base distance (given in Eq. 7).

We used 3 different clustering algorithms: a hierarchical clustering with complete linkage, a standard K-median, and a spectral algorithm by Ng et al. (2002). Since each algorithm takes a number of clusters as an input parameter, we varied the number of clusters between 3 and 5. We applied clustering algorithms to the distances $d_{CM}(cov)$, $d_{CM}(freq)$, $d_U(cov)$, and $d_U(freq)$, and compare the clusterings obtained from $d_{CM}(cov)$ against the clusterings obtained from $d_U(cov)$, and similarly compare the clusterings obtained from $d_{CM}(freq)$ against the clusterings obtained from $d_U(freq)$.

We measured the performance using 3 different clustering indices: a ratio r of the mean of the intra-cluster distances and the mean of the inter-cluster distances, Davies-Bouldin (DB) index (Davies and Bouldin, 1979), and Calinski-Harabasz (CH) index (Calinski and Harabasz, 1974).

The obtained results were studied in the following way: Given a data set and a performance index, we calculated the number of algorithms in which $d_{CM}(cov)$ outperformed $d_U(cov)$. The distances $d_{CM}(freq)$ and $d_U(freq)$ were handled similarly. The results are given in Table 3. We also calculated the number of data sets in which $d_{CM}(cov)$ outperformed $d_U(cov)$, given an algorithm and an index. These results are given in Table 4.

		$d_{CM}(cov)$ vs. $d_U(cov)$			$d_{CM}(freq)$ vs. $d_U(freq)$				
	Data set	r	DB	CH	r	DB	CH	Total	P
1.	*20Newsgroups*	0/9	2/9	7/9	8/9	5/9	9/9	31/54	0.22
2.	*Speeches*	9/9	6/9	3/9	9/9	9/9	9/9	**45/54**	0.00
3.	*Bible*	9/9	7/9	2/9	9/9	7/9	9/9	**43/54**	0.00
4.	*Beatles*	0/9	3/9	6/9	0/9	1/9	0/9	**10/54**	0.00
5.	*TopGenres*	0/9	4/9	5/9	0/9	1/9	0/9	**10/54**	0.00
6.	*TopDecades*	3/9	7/9	2/9	7/9	7/9	9/9	**35/54**	0.02
7.	*Abstract*	9/9	8/9	1/9	0/9	2/9	1/9	21/54	0.08
	Total	30/63	37/63	26/63	33/63	32/63	37/63	195/378	0.50
	P	0.61	0.13	0.13	0.61	0.80	0.13		

Table 3: Summary of the performance results of the CM distance versus the base distance. A single entry contains the number of clustering algorithm configurations (see Column 1 in Table 4) in which the CM distance was better than the base distance. The P-value is the standard Fisher's sign test.

We see from Table 3 that the performance of CM distance against the base distance depends on the data set. For example, the CM distance has tighter clusterings in *Speeches*, *Bible*, and *TopDecade* whereas the base distance outperforms the CM distance in *Beatles* and *TopGenre*.

Table 4 suggests that the overall performance of the CM distance is as good as the base distance. The CM distance obtains a better index 195 times out of 378. The statistical test suggests that this is a tie. The same observation is true if we compare the distances algorithmic-wise or index-wise.

7.4 Distance Matrices

In this section we will investigate the CM distance matrices for real-world data sets. Our goal is to demonstrate that the CM distance produces interesting and interpretable results.

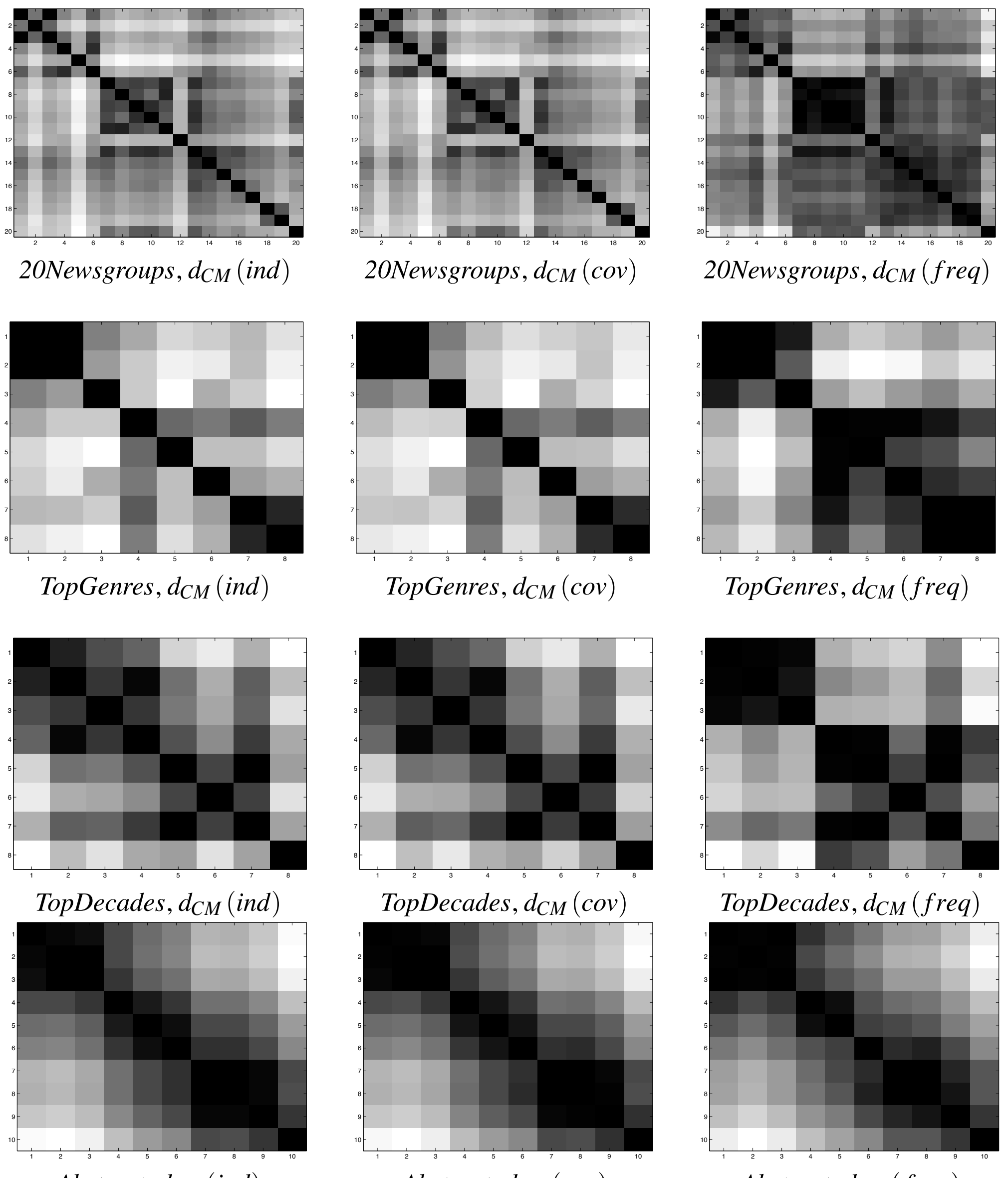
		$d_{CM}(cov)$ vs. $d_U(cov)$			$d_{CM}(freq)$ vs. $d_U(freq)$				
	Algorithm	r	DB	CH	r	DB	CH	Total	P
1.	K-MED(3)	4/7	2/7	5/7	4/7	4/7	4/7	23/42	0.44
2.	K-MED(4)	4/7	4/7	3/7	4/7	4/7	4/7	23/42	0.44
3.	K-MED(5)	4/7	4/7	3/7	4/7	4/7	4/7	23/42	0.44
4.	LINK(3)	3/7	4/7	3/7	2/7	3/7	4/7	19/42	0.44
5.	LINK(4)	3/7	4/7	3/7	4/7	3/7	4/7	21/42	0.88
6.	LINK(5)	3/7	3/7	4/7	4/7	2/7	4/7	20/42	0.64
7.	SPECT(3)	3/7	6/7	1/7	3/7	4/7	4/7	21/42	0.88
8.	SPECT(4)	3/7	4/7	3/7	4/7	4/7	4/7	22/42	0.64
9.	SPECT(5)	3/7	6/7	1/7	4/7	4/7	5/7	23/42	0.44
	Total	30/63	37/63	26/63	33/63	32/63	37/63	195/378	0.50
	P	0.61	0.13	0.13	0.61	0.80	0.13		

Table 4: Summary of the performance results of the CM distance versus the base distance. A single entry contains the number of data sets (see Column 1 in Table 3) in which the CM distance was better than the base distance. The P-value is the standard Fisher's sign test.

We calculated the distance matrices using the feature sets *ind*, *cov*, and *freq*. The matrices are given in Figures 4 and 3. In addition, we computed performance indices, a ratio of the mean of the intra-cluster distances and the mean of the inter-cluster distances, for various clusterings and compare these indices to the ones obtained from the base distances. The results are given in Table 5.

			cov		*freq*	
Data	Clustering	ind	d_{CM}	d_U	d_{CM}	d_U
Bible	Old Test. | New Test.	0.79	0.79	0.82	0.73	0.81
	Old Test. | Gospels | Epistles	0.79	0.79	0.81	0.73	0.81
Addresses	1–32 | 33–55	0.79	0.80	0.85	0.70	0.84
	1–11 | 12–22 | 23–33 | 34–44 | 45–55	0.83	0.83	0.87	0.75	0.87
Beatles	1,2,4–6 | 7–10,12–13 | 3 | 11	0.83	0.86	0.83	0.88	0.61
	1,2,4,12,13 | 5–10 | 3 | 11	0.84	0.85	0.84	0.89	0.63
20Newsgroups	Rel.,Pol. | Rest	0.76	0.77	0.67	0.56	0.62
	Rel.,Pol. | Comp., misc | Rest	0.78	0.78	0.79	0.53	0.79
TopGenres	Act.,Adv., SciFi | Rest	0.74	0.73	0.64	0.50	0.32
TopDecades	1930–1960 | 1970–2000	0.84	0.83	0.88	0.75	0.88
	1930–1950 | 1960–2000	0.88	0.88	0.98	0.57	1.06

Table 5: Statistics of various interpretable clusterings. The proportions are the averages of the intra-cluster distances divided by the averages of the inter-cluster distances. Hence small fractions imply tight clusterings.



20Newsgroups, $d_{CM}(ind)$ — *20Newsgroups*, $d_{CM}(cov)$ — *20Newsgroups*, $d_{CM}(freq)$

TopGenres, $d_{CM}(ind)$ — *TopGenres*, $d_{CM}(cov)$ — *TopGenres*, $d_{CM}(freq)$

TopDecades, $d_{CM}(ind)$ — *TopDecades*, $d_{CM}(cov)$ — *TopDecades*, $d_{CM}(freq)$

Abstract, $d_{CM}(ind)$ — *Abstract*, $d_{CM}(cov)$ — *Abstract*, $d_{CM}(freq)$

Figure 3: Distance matrices for *20Newsgroups*, *TopGenres*, *TopDecades*, and *Abstract*. In the first column the feature set *ind* contains the independent means, in the second feature set *cov* the pairwise correlation is added, and in the third column the feature set *freq* consists of $10K$ most frequent itemsets, where K is the number of attributes. Darker colours indicate smaller distances.

We should stress that standard edit distances would not work in these data setups. For example, the sequences have different lengths and hence Levenshtein distance cannot work.

The imperative observation is that, according to the CM distance, the data sets have structure. We can also provide some interpretations to the results: In *Bible* we see a cluster starting from the

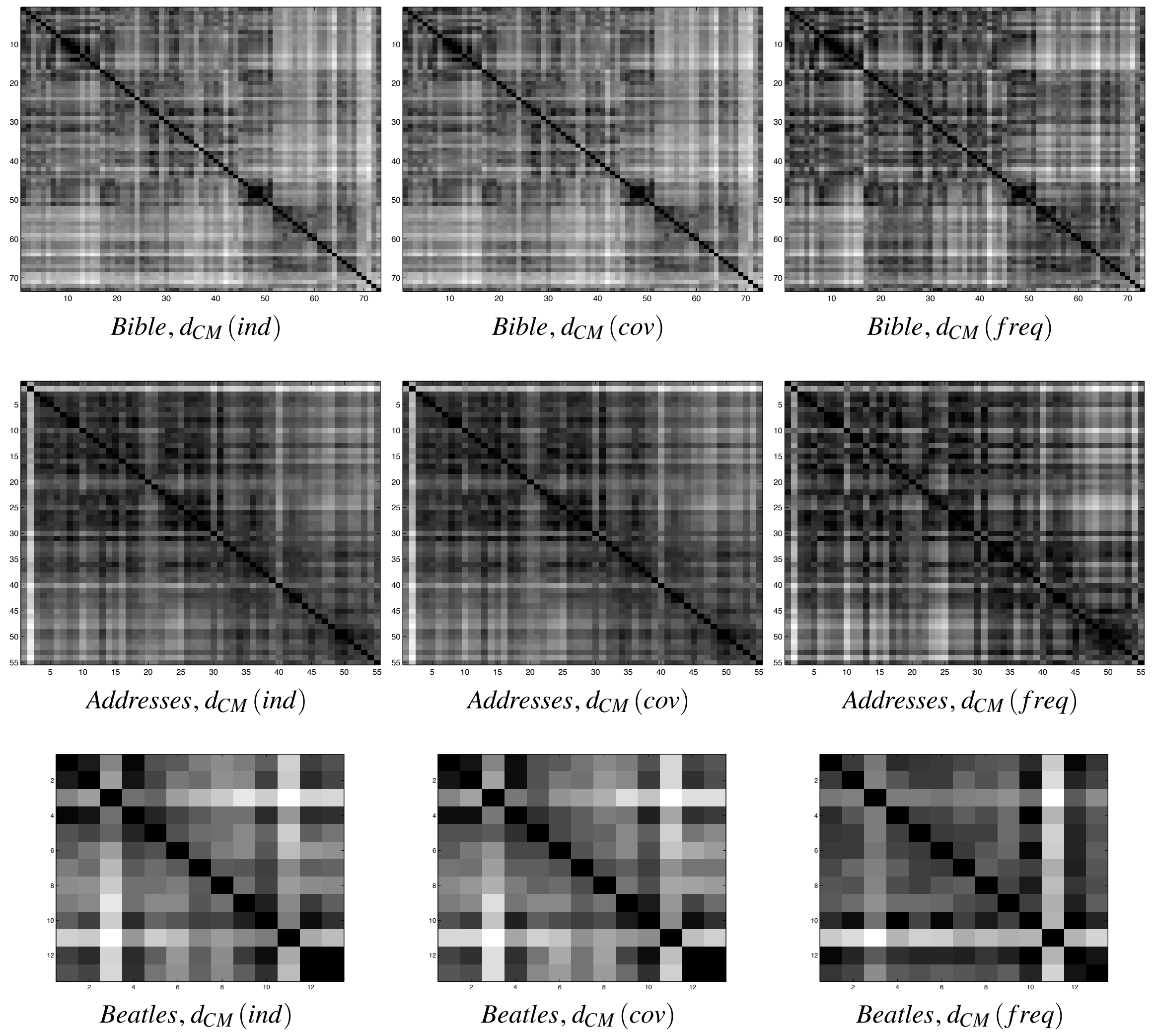


Figure 4: Distance matrices for *Bible*, *Addresses*, and *Beatles*. In the first column the feature set *ind* contains the independent means, in the second feature set *cov* the pairwise correlation is added, and in the third column the feature set *freq* consists of $10K$ most frequent itemsets, where K is the number of attributes. Darker colours indicate smaller distances.

46th book. The New Testament starts from the 47th book. An alternative clustering is obtained by separating the Epistles, starting from the 52th book, from the Gospels. In *Addresses* we some temporal dependence. Early speeches are different than the modern speeches. In *Beatles* we see that the early albums are linked together and the last two albums are also linked together. The third album, *Help!*, is peculiar. It is not linked to the early albums but rather to the later work. One explanation may be that, unlike the other early albums, this album does not contain any cover songs. In *20Newsgroups* the groups of politics and of religions are close to each other and so are the computer-related groups. The group *misc.forsale* is close to the computer-related groups. In *TopGenres Action* and *Adventure* are close to each other. Also *Comedy* and *Romance* are linked. In

TopDecades and in *Abstract* we see temporal behaviour. In Table 5 the CM distance outperforms the base distance, except for *Beatles* and *TopGenres*.

8. Conclusions and Discussion

Our task was to find a versatile distance that has nice statistical properties and that can be evaluated efficiently. The CM distance fulfils our goals. In theoretical sections we proved that this distance takes properly into account the correlation between features, and that it is the only (Mahalanobis) distance that does so. Even though our theoretical justifications are complex, the CM distance itself is rather simple. In its simplest form, it is the L_2 distance between the means of the individual attributes. On the other hand, the CM distance has a surprising form when the features are itemsets.

In general, the computation time of the CM distance depends of the size of sample space that can be exponentially large. Still, there are many types of feature functions for which the distance can be solved. For instance, if the features are itemsets, then the distance can be solved in polynomial time. In addition, if the itemsets form an antimonotonic family, then the distance can be solved in linear time.

In empirical tests the CM distance implied that the used data sets have structure, as expected. The performance of the CM distance compared to the base distance depended heavily on the data set. We also showed that the feature sets *ind* and *cov* produced almost equivalent distances, whereas using frequent itemsets produced very different distances.

Sophisticated feature selection methods were not compared in this paper. Instead, we either decided explicitly the set of features or deduced them using APRIORI. We argued that we cannot use the traditional approaches for selecting features of data sets, unless we are provided some additional information.

Acknowledgments

The author would like to thank Heikki Mannila and Kai Puolamäki for their extremely helpful comments.

Appendix A.

In this section we will prove the theorems given in this paper.

A.1 Proof of Theorem 1

To simplify the notation denote $S_0(x) = 1$, $\theta_1^* = [1, \theta_{11}, \ldots, \theta_{1N}]^T$ and $\theta_2^* = [1, \theta_{21}, \ldots, \theta_{2N}]^T$. The norm function restricted to the affine space has one minimum and it can be found using Lagrange multipliers. Thus we can express the vectors u_i in Eq. 2

$$u_{ij} = \lambda_i^T S(j),$$

where $j \in \Omega$ and λ_i is the column vector of length $N+1$ consisting of the corresponding Lagrange multipliers. The distance is equal to

$$\begin{aligned} d_{CM}(D_1, D_2 \mid S)^2 &= |\Omega| \, \|u_1 - u_2\|_2^2, \\ &= |\Omega| \sum_{j \in \Omega} (u_{1j} - u_{2j})(u_{1j} - u_{2j}), \\ &= |\Omega| \sum_{j \in \Omega} (u_{1j} - u_{2j}) \left(\lambda_1^T S(j) - \lambda_2^T S(j)\right), \\ &= |\Omega| (\lambda_1 - \lambda_2)^T \sum_{j \in \Omega} (u_{1j} - u_{2j}) S(j), \\ &= |\Omega| (\lambda_1 - \lambda_2)^T (\theta_1^* - \theta_2^*). \end{aligned}$$

Since $u_i \in \mathcal{C}(S, \theta_i)$, the multipliers λ_i can be solved from the equation

$$\theta_i^* = \sum_{j \in \Omega} S(j) u_{ij} = \sum_{j \in \Omega} S(j) \lambda_i^T S(j) = \left(\sum_{j \in \Omega} S(j) S(j)^T \right) \lambda_i,$$

that is, $\theta_i^* = A\lambda_i$, where A is an $(N+1) \times (N+1)$ matrix $A_{xy} = \sum_j S_x(j) S_y(j)$. It is straightforward to prove that the existence of $\operatorname{Cov}^{-1}[S]$ implies that A is also invertible. Let B be an $N \times N$ matrix formed from A^{-1} by removing the first row and the first column. We have

$$\begin{aligned} |\Omega| \, \|u_1 - u_2\|_2^2 &= |\Omega| (\theta_1^* - \theta_2^*)^T A^{-1} (\theta_1^* - \theta_2^*), \\ &= |\Omega| (\theta_1 - \theta_2)^T B (\theta_1 - \theta_2). \end{aligned}$$

The last equality is true since $\theta_{10}^* = \theta_{20}^*$.

We need to prove that $|\Omega| B = \operatorname{Cov}^{-1}[S]$. Let $[c; B]$ be the matrix obtained from A^{-1} by removing the first row. Let $\gamma = \mathrm{E}[S]$ taken with respect to the uniform distribution. Since the first column of A is equal to $|\Omega| [1, \gamma]$, it follows that $c = -B\gamma$. From the identity

$$c_x A_{(0,y)} + \sum_{z=1}^{N} B_{(x,z)} A_{(z,y)} = \delta_{xy}$$

we have

$$\sum_{z=1}^{N} B_{(x,z)} \left(A_{(z,y)} - A_{(0,y)} \gamma_z\right) = \sum_{z=1}^{N} |\Omega| B_{(x,z)} \left(|\Omega|^{-1} A_{(z,y)} - \gamma_y \gamma_z \right) = \delta_{xy}.$$

Since $|\Omega|^{-1} A_{(z,y)} - \gamma_z \gamma_y$ is equal to the (z, y) entry of $\operatorname{Cov}[S]$, the theorem follows.

A.2 Proofs of Theorems given in Section 2.2

Proof [Theorem 2] The covariance matrix $\operatorname{Cov}[S]$ in Theorem 1 depends only on S and is positive definite. Therefore, the CM distance is a Mahalanobis distance. ■

Proof [Theorem 3] Let $\theta_i = S(D_i)$ for $i = 1, 2, 3$. The frequencies for $D_1 \cup D_3$ and $D_2 \cup D_3$ are $(1-\varepsilon)\theta_1 + \varepsilon\theta_3$ and $(1-\varepsilon)\theta_2 + \varepsilon\theta_3$, respectively. The theorem follows from Theorem 1. ■

The following lemma proves Theorem 4.

Lemma 10 *Let $A : \mathbb{R}^N \to \mathbb{R}^M$ and define a function $T(\omega) = A(S(\omega))$. Let $\phi = T(D)$ and $\theta = S(D)$ be the frequencies for some data set D. Assume further that there is no two data sets D_1 and D_2 such that $S(D_1) = S(D_2)$ and $T(D_1) \neq T(D_2)$. Then $d_{CM}(D_1, D_2 \mid T) \leq d_{CM}(D_1, D_2 \mid S)$. The equality holds if for a fixed ϕ the frequency θ is unique.*

Before proving this lemma, let us explain why the uniqueness requirement is needed: Assume that the sample space Ω consists of two-dimensional binary vectors, that is,

$$\Omega = \{(0,0), (1,0), (0,1), (1,1)\}.$$

We set the features to be $S(\omega) = [\omega_1, \omega_2]^T$. Define a function $T(x) = [\omega_1, \omega_2, \omega_1\omega_2]^T = [S_1(\omega), S_2(\omega), S_1(\omega)S_2(\omega)]^T$. Note that uniqueness assumption is now violated. Without this assumption the lemma would imply that $d_{CM}(D_1, D_2 \mid T) \leq d_{CM}(D_1, D_2 \mid S)$ which is in general false.
Proof Let $\theta_1 = S(D_1)$ and $\phi_1 = T(D_1)$. Pick $u \in \mathcal{C}(S, \theta_1)$. The frequency of S taken with the respect to u is θ_1 and because of the assumption the corresponding frequency of T is ϕ_1. It follows that $\mathcal{C}(S, \theta_i) \subseteq \mathcal{C}(T, \phi_i)$. The theorem follows from the fact that the CM distance is the shortest distance between the affine spaces $\mathcal{C}(S, \theta_1)$ and $\mathcal{C}(S, \theta_2)$. ■

A.3 Proof of Theorem 7

It suffices to prove that the matrix $C(S)$ is proportional to the covariance matrix $\operatorname{Cov}[S]$. The notation $\delta(\omega_1 \mid \omega_2)$ used in the proof represents a feature function $\delta : \Omega \to \{0, 1\}$ which returns 1 if $\omega_1 = \omega_2$ and 0 otherwise.

Before proving the theorem we should point one technical detail. In general, $C(S)$ may be singular, especially in Assumption 1. In our proof we will show that $C(S) \propto \operatorname{Cov}[S]$ and this does not require $C(S)$ to be invertible. However, if one wants to evaluate the distance d, then one must assume that $C(S)$ is invertible.

Fix indices i and j such that $i \neq j$. Let $T(\omega) = [S_i(\omega), S_j(\omega)]^T$. If follows from Assumption 1 that

$$C(T) = \begin{bmatrix} C_{ii}(S) & C_{ij}(S) \\ C_{ji}(S) & C_{jj}(S) \end{bmatrix}.$$

This implies that $C_{ij}(S)$ depends only on S_i and S_j. In other words, we can say $C_{ij}(S) = C_{ij}(S_i, S_j)$. Let $\rho : \{1, \ldots, N\} \to \{1, \ldots, N\}$ be some permutation function and define $U(x) = \left[S_{\rho(1)}(x), \ldots, S_{\rho(N)}(x)\right]^T$. Assumption 1 implies that

$$C_{\rho(i)\rho(j)}(S) = C_{ij}(U) = C_{ij}(U_i, U_j) = C_{ij}(S_{\rho(i)}, S_{\rho(j)}).$$

This is possible only if all non-diagonal entries of C have the same form or, in other words, $C_{ij}(S) = C_{ij}(S_i, S_j) = C(S_i, S_j)$. Similarly, the diagonal entry S_{ii} depends only on S_i and all the diagonal entries have the same form $C_{ii}(S) = C(S_i)$. To see the connection between $C(S_i)$ and $C(S_i, S_j)$ let $V(\omega) = [S_i(\omega), S_i(\omega)]^T$ and let $W(\omega) = [2S_i(\omega)]^T$. We can represent $W(\omega) = V_1(\omega) + V_2(\omega)$. Now Assumption 1 implies

$$\begin{aligned} 4C(S_i) = C(W) &= C(V_{11}) + 2C(V_{12}, V_{21}) + C(V_{22}), \\ &= 2C(S_i) + 2C(S_i, S_i) \end{aligned}$$

which shows that $C(S_i) = C(S_i, S_i)$. Fix S_j and note that Assumption 1 implies that $C(S_i, S_j)$ is a linear function of S_i. Thus C has a form

$$C(S_i, S_j) = \sum_{\omega \in \Omega} S_i(\omega) h(S_j, \omega)$$

for some specific map h. Let $\alpha \in \Omega$. Then $C(\delta(\omega \mid \alpha), S_j) = h(S_j, \alpha)$ is a linear function of S_j. Thus C has a form

$$C(S_i, S_j) = \sum_{\omega_1, \omega_2 \in \Omega} S_i(\omega_1) S_j(\omega_2) g(\omega_1, \omega_2)$$

for some specific g.

Let α, β, and γ be distinct points in Ω. An application of Assumption 2 shows that $g(\alpha, \beta) = C(\delta(\omega \mid \alpha), \delta(\omega \mid \beta)) = C(\delta(\omega \mid \alpha), \delta(\omega \mid \gamma)) = g(\alpha, \gamma)$. Thus g has a form $g(\omega_1, \omega_2) = a\delta(\omega_1 \mid \omega_2) + b$ for some constants a and b.

To complete the proof note that Assumption 1 implies that $C(S + b) = C(S)$ which in turns implies that $\sum_x g(\omega_1, \omega_2) = 0$ for all y. Thus $b = -a \left|\Omega\right|^{-1}$. This leads us to

$$\begin{aligned} C(S_i, S_j) &= \sum_{\omega_1, \omega_2 \in \Omega} S_i(\omega_1) S_j(\omega_2) \left(a\delta(\omega_1 \mid \omega_2) - a \left|\Omega\right|^{-1} \right), \\ &= a \sum_{\omega \in \Omega} S_i(\omega) S_j(\omega) - a \left(\sum_{\omega \in \Omega} S_i(\omega) \right) \left(\sum_{\omega \in \Omega} \left|\Omega\right|^{-1} S_j(\omega) \right), \\ &\propto \mathrm{E}\left[S_i S_j\right] - \mathrm{E}\left[S_i\right] \mathrm{E}\left[S_j\right], \end{aligned}$$

where the means are taken with respect to the uniform distribution. This identity proves the theorem.

A.4 Proof for Lemma 8

Let us prove that $\mathrm{Cov}\left[T_{\mathcal{F}}\right] = 0.5I$. Let A be an itemset. There are odd number of ones in A in exactly half of the transactions. Hence, $\mathrm{E}\left[T_A^2\right] = \mathrm{E}\left[T_A\right] = 0.5$. Let $B \neq A$ be an itemset. We wish to have $T_B(\omega) = T_A(\omega) = 1$. This means that ω must have odd number of ones in A and in B. Assume that the number of ones in $A \cap B$ is even. This means that $A - B$ and $B - A$ have odd number of ones. There is only a quarter of all the transactions that fulfil this condition. If $A \cap B$ is odd, then we must an even number of ones in $A - B$ and $B - A$. Again, there is only a quarter of all the transactions for which this holds. This implies that $\mathrm{E}\left[T_A T_B\right] = 0.25 = \mathrm{E}\left[T_A\right] \mathrm{E}\left[T_B\right]$. This proves that $\mathrm{Cov}\left[T_{\mathcal{F}}\right] = 0.5I$.

A.5 Proof of Theorem 9

Before proving this theorem let us rephrase it. First, note even though $d(\cdot, \cdot \mid \cdot)$ is defined only on the conjunction functions $S_{\mathcal{F}}$, we can operate with the parity function $T_{\mathcal{F}}$. As we stated before there is an invertible matrix A such that $T_{\mathcal{F}} = AS_{\mathcal{F}}$. We can write the distance as

$$d(D_1, D_2 \mid S_{\mathcal{F}})^2 = (A\theta_1 - A\theta_2)^T \left(A^{-1}\right)^T C(S_{\mathcal{F}})^{-1} A^{-1} (A\theta_1 - A\theta_2).$$

Thus we define $C(T_{\mathcal{F}}) = AC(S_{\mathcal{F}})A^T$. Note that the following lemma implies that the condition stated in Theorem 9 is equivalent to $C(T_{\mathcal{A}}) = cI$, for some constant c. Theorem 9 is equivalent to stating that $C(T_{\mathcal{F}}) = cI$.

The following lemma deals with some difficulties due the fact that the frequencies should arise from some valid distributions

Lemma 11 *Let $\mathcal{A}$ be the family of all itemsets. There exists $\varepsilon > 0$ such that for each real vector γ of length $2^K - 1$ that satisfies $\|\gamma\|_2 < \varepsilon$ there exist distributions p and q such that $\gamma = \mathrm{E}_p[T_\mathcal{A}] - \mathrm{E}_q[T_\mathcal{A}]$.*

Proof To ease the notation, add $T_0(x) = 1$ to $T_\mathcal{A}$ and denote the end result by T^*. We can consider T^* as a $2^K \times 2^K$ matrix, say A. Let p be a distribution and let u be the vector of length 2^K representing the distribution. Note that we have $Au = \mathrm{E}_p[T^*]$. We can show that A is invertible. Let U some $2^K - 1$ dimensional open ball of distributions. Since A is invertible, the set $V^* = \{Ax \mid x \in U\}$ is a $2^K - 1$ dimensional open ellipsoid. Define also V by removing the first coordinate from the vectors of V^*. Note that the first coordinate of elements of V^* is equal to 1. This implies that V is also a $2^K - 1$ dimensional open ellipsoid. Hence we can pick an open ball $N(\theta, \varepsilon) \subset V$. The lemma follows. ■

We are now ready to prove Theorem 9:

Abbreviate the matrix $C(T_\mathcal{F})$ by C. We will first prove that the diagonal entries of C are equal to c. Let $\mathcal{A}$ be the family of all itemsets. Select $G \in \mathcal{F}$ and define $\mathcal{R} = \{H \in \mathcal{F} \mid H \subseteq G\}$. As we stated above, $C(T_\mathcal{A}) = cI$ and Assumption 2 imply that $C(T_\mathcal{R}) = cI$. Assumption 1 implies that

$$d\left(\cdot,\cdot \mid S_\mathcal{R}\right)^2 \leq d\left(\cdot,\cdot \mid S_\mathcal{F}\right)^2 \leq d\left(\cdot,\cdot \mid S_\mathcal{A}\right)^2. \tag{8}$$

Select ε corresponding to Lemma 11 and let $\gamma_\mathcal{A} = [0,\ldots,\varepsilon/2,\ldots,0]^T$, that is, $\gamma_\mathcal{A}$ is a vector whose entries are all 0 except the entry corresponding to G. Lemma 11 guarantees that there exist distributions p and q such that $d(p,q \mid S_\mathcal{A})^2 = c\,\|\gamma_\mathcal{A}\|_2^2$. Let $\gamma_\mathcal{F} = \mathrm{E}_p[T_\mathcal{F}] - \mathrm{E}_q[T_\mathcal{F}]$ and $\gamma_\mathcal{R} = \mathrm{E}_p\left[T_\mathcal{R}\right] - \mathrm{E}_q\left[T_\mathcal{R}\right]$. Note that $\gamma_\mathcal{R}$ and $\gamma_\mathcal{F}$ has the same form as γ_A. It follows from Eq. 8 that

$$c\varepsilon^2/4 \leq C_{G,G}\varepsilon^2/4 \leq c\varepsilon^2/4,$$

where $C_{G,G}$ is the diagonal entry of C corresponding to G. It follows that $C_{G,G} = c$.

To complete the proof we need to show that $C_{G,H} = 0$ for $G,H \in \mathcal{F}, G \neq H$. Assume that $C_{X,Y} \neq 0$ and let s be the sign of $C_{G,H}$. Apply Lemma 11 again and select $\gamma_\mathcal{A} = [0,\ldots,\varepsilon/4,0,\ldots,0,s\varepsilon/4,\ldots,0]^T$, that is, $\gamma_\mathcal{A}$ has $\varepsilon/4$ and $s\varepsilon/4$ in the entries corresponding to G and H, respectively, and 0 elsewhere. The right side of Eq. 8 implies that

$$2c\varepsilon^2/16 + 2\left|C_{G,H}\right|\varepsilon^2/16 \leq 2c\varepsilon^2/16$$

which is a contradiction and it follows that $C_{G,H} = 0$. This completes the theorem.

References

Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 26–28 1993.

Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and Aino I. Verkamo. Fast discovery of association rules. In Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI Press/The MIT Press, 1996.

Pierre Baldi, Paolo Frasconi, and Padhraic Smyth. *Modeling the Internet and the Web*. John Wiley & Sons, 2003.

Michéle Baseville. Distance measures for signal processing and pattern recognition. *Signal Processing*, 18(4):349–369, 1989.

Toon Calders. *Axiomatization and Deduction Rules for the Frequency of Itemsets*. PhD thesis, University of Antwerp, Belgium, 2003.

Tadeusz Calinski and Jerzy Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3:1–27, 1974.

Gregory Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42(2–3):393–405, Mar. 1990.

Imre Csiszár. I-divergence geometry of probability distributions and minimization problems. *The Annals of Probability*, 3(1):146–158, Feb. 1975.

David L. Davies and Donald W. Bouldin. A cluster separation measure. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 1(2):224–227, April 1979.

Thomas Eiter and Heikki Mannila. Distance measures for point sets and their computation. *Acta Informatica*, 34(2):109–133, 1997.

Theodore Hailperin. Best possible inequalities for the probability of a logical function of events. *The American Mathematical Monthly*, 72(4):343–359, Apr. 1965.

David Hand, Heikki Mannila, and Padhraic Smyth. *Principles of Data Mining*. The MIT Press, 2001.

Jaakko Hollmén, Jouni K Seppänen, and Heikki Mannila. Mixture models and frequent sets: Combining global and local methods for 0-1 data. In *Proceedings of the SIAM Conference on Data Mining (2003)*, 2003.

Solomon Kullback. *Information Theory and Statistics*. Dover Publications, Inc., 1968.

Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady.*, 10(8):707–710, February 1966.

Heikki Mannila, Dmitry Pavlov, and Padhraic Smyth. Prediction with local patterns using cross-entropy. In *Knowledge Discovery and Data Mining*, pages 357–361, 1999.

Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, 2002.

Judea Pearl. *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

Journal of Machine Learning Research 8 (2007) 155-201 Submitted 11/04; Revised 12/05; Published 1/07

Building Blocks for Variational Bayesian Learning of Latent Variable Models

Tapani Raiko TAPANI.RAIKO@HUT.FI
Harri Valpola HARRI.VALPOLA@HUT.FI
Markus Harva MARKUS.HARVA@HUT.FI
Juha Karhunen JUHA.KARHUNEN@HUT.FI
Adaptive Informatics Research Centre
Helsinki University of Technology
P.O.Box 5400, FI-02015 HUT, Espoo, Finland

Editor: Michael I. Jordan

Abstract

We introduce standardised building blocks designed to be used with variational Bayesian learning. The blocks include Gaussian variables, summation, multiplication, nonlinearity, and delay. A large variety of latent variable models can be constructed from these blocks, including nonlinear and variance models, which are lacking from most existing variational systems. The introduced blocks are designed to fit together and to yield efficient update rules. Practical implementation of various models is easy thanks to an associated software package which derives the learning formulas automatically once a specific model structure has been fixed. Variational Bayesian learning provides a cost function which is used both for updating the variables of the model and for optimising the model structure. All the computations can be carried out locally, resulting in linear computational complexity. We present experimental results on several structures, including a new hierarchical nonlinear model for variances and means. The test results demonstrate the good performance and usefulness of the introduced method.

Keywords: latent variable models, variational Bayesian learning, graphical models, building blocks, Bayesian modelling, local computation

1. Introduction

Various generative modelling approaches have provided powerful statistical learning methods for neural networks and graphical models during the last years. Such methods aim at finding an appropriate model which explains the internal structure or regularities found in the observations. It is assumed that these regularities are caused by certain latent variables (also called factors, sources, hidden variables, or hidden causes) which have generated the observed data through an unknown mapping (Bishop, 1999). In unsupervised learning, the goal is to identify both the unknown latent variables and generative mapping.

The expectation-maximisation (EM) algorithm (Dempster et al., 1977) has often been used for learning latent variable models. The distribution for the latent variables is modelled, but the model parameters are found using maximum likelihood or maximum a posteriori estimators. However, with such point estimates, determination of the correct model order and overfitting are ubiquitous and often difficult problems. Therefore, full Bayesian approaches making use of the complete posterior distribution have recently gained a lot of attention. Exact treatment of the posterior distribution is intractable except in simple toy problems, and hence one must resort to suitable approximations. So-called Laplace approximation (Tierney and Kadane, 1986; Swartz and Evans, 2000) employs a Gaussian approximation around the peak of the posterior distribution. However, this method still suffers from overfitting. In real-world problems, it often does not perform adequately, and has therefore largely given way for better alternatives. Among them, Markov Chain Monte Carlo (MCMC) techniques (Gelfand and Smith, 1990; Geman and Geman, 1984; Swartz and Evans, 2000) are popular in supervised learning tasks, providing good estimation results. Unfortunately, the computational load is high, which restricts the use of MCMC in large scale unsupervised learning problems where the parameters and variables to be estimated are numerous. For instance, Rowe (2003) has a case study in unsupervised learning from brain imaging data. He used MCMC for a scaled down toy example but resorted to point estimates with real data.

Variational Bayesian learning (Wallace, 1990; Hinton and van Camp, 1993; Neal and Hinton, 1999; Waterhouse et al., 1996; Lappalainen and Miskin, 2000) (for an introduction, see Jordan et al., 1999; MacKay, 2003) has gained increasing attention during the last years. This is because it largely avoids overfitting (Honkela and Valpola, 2004), allows for estimation of the model order and structure, and its computational load is reasonable compared to the MCMC methods. Variational Bayesian learning was first employed in supervised problems (Wallace, 1990; Hinton and van Camp, 1993; MacKay, 1995; Barber and Bishop, 1998), but it has now become popular also in unsupervised modelling. Recently, several authors have successfully applied such techniques to linear factor analysis, independent component analysis (ICA) (Hyvärinen et al., 2001; Roberts and Everson, 2001; Choudrey et al., 2000; Højen-Sørensen et al., 2002), and their various extensions. These include linear independent factor analysis (Attias, 1999), several other extensions of the basic linear ICA model (Attias, 2001; Chan et al., 2001; Miskin and MacKay, 2001; Roberts et al., 2004), mixture models and other graphical models (Attias, 2000; Beal and Ghahramani, 2003; Winn and Bishop, 2005), as well as MLP networks for modelling nonlinear observation mappings (Lappalainen and Honkela, 2000; Hyvärinen et al., 2001) and nonlinear dynamics of the latent variables (source signals) (Ilin et al., 2003; Valpola and Karhunen, 2002; Valpola et al., 2003b). Variational Bayesian learning has also been applied to large discrete models (Murphy, 1999) such as sigmoid belief networks (Jaakkola and Jordan, 1997), nonlinear belief networks (Frey and Hinton, 1999), and hidden Markov models (MacKay, 1997).

In this paper, we introduce a small number of basic blocks for building latent variable models which are learned using variational Bayesian learning. The blocks have been introduced earlier in two conference papers (Valpola et al., 2001; Harva et al., 2005) and their applications in Valpola et al. (2003c), Honkela et al. (2003), Honkela et al. (2005), Raiko et al. (2003), Raiko (2004), and Raiko (2005). Valpola et al. (2004) studied hierarchical models for variance sources from signal processing point of view. This paper is the first comprehensive presentation about the block framework itself. Our approach is most suitable for unsupervised learning tasks which are considered in this paper, but in principle at least, it could be applied to supervised learning, too. A wide variety of factor analysis type latent variable models can be constructed by combining the basic blocks suitably. Variational Bayesian learning then provides a cost function which can be used for updating the variables as well as for optimising the model structure. The blocks are designed so as to fit together and yield efficient update rules. By using a maximally factorial posterior approximation, all the required computations can be performed locally. This results in linear computational complexity as a function of the number of connections in the model. The Bayes Blocks software package by Valpola et al. (2003a) is an open-source C++/Python implementation that can freely be downloaded.

The basic building block is a Gaussian variable (node). It uses as its input values both mean and variance. The other building blocks include addition and multiplication nodes, delay, and a Gaussian variable followed by a nonlinearity. Several known model structures can be constructed using these blocks. We also introduce some novel model structures by extending known linear structures using nonlinearities and variance modelling. Examples will be presented later on in this paper.

The key idea behind developing these blocks is that after the connections between the blocks in the chosen model have been fixed (that is, a particular model has been selected and specified), the cost function and the updating rules needed in learning can be computed automatically. The user does not need to understand the underlying mathematics since the derivations are done within the software package. This allows for rapid prototyping. The Bayes Blocks can also be used to bring different methods into a unified framework, by implementing a corresponding structure from blocks and by using results of these methods for initialisation. Different methods can then be compared directly using the cost function and perhaps combined to find even better models. Updates that minimise a global cost function are guaranteed to converge, unlike algorithms such as loopy belief propagation (Pearl, 1988), extended Kalman smoothing (Anderson and Moore, 1979), or expectation propagation (Minka, 2001).

Winn and Bishop (2005) have introduced a general purpose algorithm called variational message passing. It resembles our framework in that it uses variational Bayesian learning and factorised approximations. The VIBES framework allows for discrete variables but not nonlinearities or nonstationary variance. The posterior approximation does not need to be fully factorised which leads to a more accurate model. Optimisation proceeds by cycling through each factor and revising the approximate posterior distribution. Messages that contain certain expectations over the posterior approximation are sent through the network.

Beal and Ghahramani (2003); Ghahramani and Beal (2001), and Beal (2003) view variational Bayesian learning as an extension to the EM algorithm. Their algorithms apply to combinations of discrete and linear Gaussian models. In the experiments, the variational Bayesian model structure selection outperformed the Bayesian information criterion (Schwarz, 1978) at relatively small computational cost, while being more reliable than annealed importance sampling (Neal, 2001) even with the number of samples so high that the computational cost is hundredfold.

A major difference of our approach compared to the related methods by Winn and Bishop (2005) and by Beal and Ghahramani (2003) is that they concentrate mainly on situations where there is a handy conjugate prior (Gelman et al., 1995) available. This restriction makes computations easier, but on the other hand our blocks can be combined more freely, allowing richer model structures. For instance, the modelling of variance in a way described in Section 5.1, would not be possible using the gamma distribution for the precision parameter in the Gaussian node. The price we have to pay for this advantage is that the minimum of the cost function must be found iteratively, while it can be solved analytically when conjugate distributions are applied. The cost function can always be evaluated analytically in the Bayes Blocks framework as well. Note that the different approaches would fit together.

Similar graphical models can be learned with sampling based algorithms instead of variational Bayesian learning. For instance, the BUGS software package by Spiegelhalter et al. (1995) uses Gibbs sampling for Bayesian inference. It supports mixture models, nonlinearities, and nonstationary variance. There are also many software packages concentrated on discrete Bayesian networks. Notably, the Bayes Net toolbox by Murphy (2001) can be used for Bayesian learning and inference of many types of directed graphical models using several methods. It also includes decision-theoretic nodes. Hence it is in this sense more general than our work. A limitation of the Bayes net toolbox (Murphy, 2001) is that it supports latent continuous nodes only with Gaussian or conditional Gaussian distributions.

Autobayes (Gray et al., 2002) is a system that generates code for efficient implementations of algorithms used in Bayes networks. Currently the algorithm schemas include EM, k-means, and discrete model selection. This system does not yet support continuous hidden variables, nonlinearities, variational methods, MCMC, or temporal models. One of the greatest strengths of the code generation approach compared to a software library is the possibility of automatically optimising the code using domain information.

In the independent component analysis (ICA, see textbook by Hyvärinen et al., 2001) community, traditionally, the observation noise has not been modelled in any way. Even when it is modelled, the noise variance is assumed to have a constant value which is estimated from the available observations when required. However, more flexible variance models would be highly desirable in a wide variety of situations. It is well-known that many real-world signals or data sets are nonstationary, being roughly stationary on fairly short intervals only. Quite often the amplitude level of a signal varies markedly as a function of time or position, which means that its variance is nonstationary. Examples include financial data sets, speech signals, and natural images.

Recently, Parra, Spence, and Sajda (2001) have demonstrated that several higher-order statistical properties of natural images and signals are well explained by a stochastic model in which an otherwise stationary Gaussian process has a nonstationary variance. Variance models are also useful in explaining volatility of financial time series and in detecting outliers in the data. On certain conditions the nonstationarity of variance makes it possible to perform blind source separation (Hyvärinen et al., 2001; Pham and Cardoso, 2001).

Several authors have introduced hierarchical models related to those discussed in this paper. These models use subspaces of dependent features instead of single feature components. This kind of models have been proposed at least in context with independent component analysis (Cardoso, 1998; Hyvärinen and Hoyer, 2000b,a; Park and Lee, 2004), and topographic or self-organising maps (Kohonen et al., 1997; Ghahramani and Hinton, 1998). A problem with these methods is that it is difficult to learn the structure of the model or to compare different model structures.

The remainder of this paper is organised as follows. In the following section, we briefly present basic concepts of variational Bayesian learning. In Section 3, we introduce the building blocks (nodes), and in Section 4 we discuss variational Bayesian computations with them. In the next section, we show examples of different types of models which can be constructed using the building blocks. Section 6 deals with learning and potential problems related with it, and in Section 7 we present experimental results on several structures given in Section 5. This is followed by a short discussion as well as conclusions in the last section of the paper.

2. Variational Bayesian Learning

In Bayesian data analysis and estimation methods (Gelman et al., 1995),all the uncertain quantities are modelled in terms of their joint probability density function (pdf). The key principle is to construct the joint posterior pdf for all the unknown quantities in a model, given the data sample. This posterior density contains all the relevant information on the unknown variables and parameters.

Denote by θ the set of all parameters for the model $\mathcal{H}$ and other unknown variables that we wish to estimate from a given data set X. The posterior probability density $p(\theta \mid X, \mathcal{H})$ of the parameters

θ given the data X is obtained from Bayes rule[1]

$$p(\theta \mid X, \mathcal{H}) = \frac{p(X \mid \theta, \mathcal{H}) p(\theta \mid \mathcal{H})}{p(X \mid \mathcal{H})}. \tag{1}$$

Here $p(X \mid \theta, \mathcal{H})$ is the likelihood of the parameters θ given the data X, $p(\theta \mid \mathcal{H})$ is the prior pdf of these parameters, and

$$p(X \mid \mathcal{H}) = \int_\theta p(X \mid \theta, \mathcal{H}) p(\theta \mid \mathcal{H}) d\theta$$

is a normalising term which is called the evidence. The evidence can be directly understood as the marginal probability of the observed data X assuming the chosen model $\mathcal{H}$. By evaluating the evidences $p(X \mid \mathcal{H}_i)$ for different models $\mathcal{H}_i$, one can therefore choose the model which describes the observed data with the highest probability (See Valpola and Karhunen, 2002; Bishop, 1995; MacKay, 2003, for a more complete discussion).

Variational Bayesian learning (Wallace, 1990; Hinton and van Camp, 1993; Neal and Hinton, 1999; Waterhouse et al., 1996; Barber and Bishop, 1998; Lappalainen and Miskin, 2000) is a fairly recently introduced approximate fully Bayesian method, which has become popular. Its key idea is to approximate the exact posterior distribution $p(\theta \mid X, \mathcal{H})$ by another distribution $q(\theta)$ that is computationally easier to handle. The approximating distribution is usually chosen to be a product of several independent distributions, one for each parameter or a set of similar parameters.

Variational Bayesian learning employs the Kullback-Leibler (KL) information (divergence) between two probability distributions $q(v)$ and $p(v)$. The KL information is defined by the cost function

$$\mathcal{J}_{\mathrm{KL}}(q \parallel p) = \int_v q(v) \ln \frac{q(v)}{p(v)} dv \tag{2}$$

which measures the difference in the probability mass between the densities $q(v)$ and $p(v)$. Its minimum value 0 is achieved when the densities $q(v)$ and $p(v)$ are the same.

The KL information is used to minimise the misfit between the actual posterior pdf $p(\theta \mid X, \mathcal{H})$ and its parametric approximation $q(\theta)$. However, the exact KL information $\mathcal{J}_{\mathrm{KL}}(q(\theta) \parallel p(\theta \mid X, \mathcal{H}))$ between these two densities does not yet yield a practical cost function, because the normalising term $p(X \mid \mathcal{H})$ needed in computing $p(\theta \mid X, \mathcal{H})$ cannot usually be evaluated.

Therefore, the cost function used in variational Bayesian learning is defined (Hinton and van Camp, 1993; MacKay, 1995)

$$\mathcal{C}_{\mathrm{KL}} = \mathcal{J}_{\mathrm{KL}}(q(\theta) \parallel p(\theta \mid X, \mathcal{H})) - \ln p(X \mid \mathcal{H}). \tag{3}$$

After slight manipulation, this yields

$$\mathcal{C}_{\mathrm{KL}} = \int_\theta q(\theta) \ln \frac{q(\theta)}{p(X, \theta \mid \mathcal{H})} d\theta \tag{4}$$

which does not require $p(X \mid \mathcal{H})$ any more. The cost function $\mathcal{C}_{\mathrm{KL}} = \mathcal{C}_q + \mathcal{C}_p$ consists of two parts, the negative entropy $\mathcal{C}_q$ and the expected energy $\mathcal{C}_p$:

$$\mathcal{C}_q = \langle \ln q(\theta) \rangle = \int_\theta q(\theta) \ln q(\theta) d\theta, \tag{5}$$

1. The subscripts of all pdf's are assumed to be the same as their arguments, and are omitted for keeping the notation simpler.

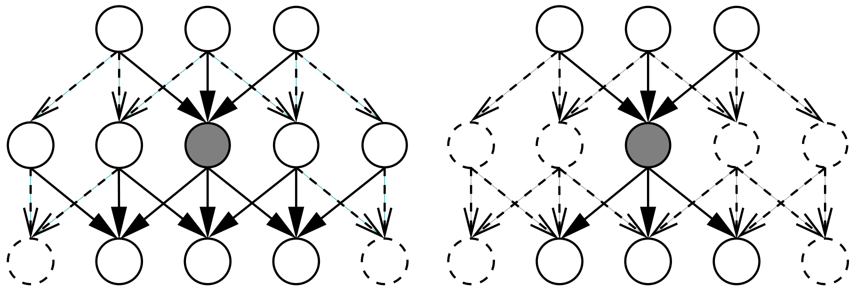


Figure 1: The dash-lined nodes and connections can be ignored while updating the shadowed node. *Left:* In general, the whole Markov blanket needs to be considered. *Right:* A completely factorial posterior approximation with no multiple computational paths leads to a decoupled problem. The nodes can be updated locally.

$$\mathcal{C}_p = \left\langle -\ln p(X,\theta \mid \mathcal{H}) \right\rangle = -\int_\theta q(\theta) \ln p(X,\theta \mid \mathcal{H}) d\theta \tag{6}$$

where the shorthand notation $\langle \cdot \rangle$ denotes expectation with respect to the approximate pdf $q(\theta)$.

In addition, the cost function $\mathcal{C}_{\mathrm{KL}}$ provides a bound for the evidence $p(X \mid \mathcal{H})$. Since $\mathcal{J}_{\mathrm{KL}}(q \parallel p)$ is always nonnegative, it follows directly from (3) that

$$\mathcal{C}_{\mathrm{KL}} \geq -\ln p(X \mid \mathcal{H}).$$

This shows that the negative of the cost function bounds the log-evidence from below.

It is worth noting that variational Bayesian ensemble learning can be derived from information-theoretic minimum description length coding as well (Hinton and van Camp, 1993). Further considerations on such arguments, helping to understand several common problems and certain aspects of learning, have been presented in a recent paper (Honkela and Valpola, 2004).

The dependency structure between the parameters in our method is the same as in Bayesian networks (Pearl, 1988; Bishop, 2006). Variables are seen as nodes of a graph. Each variable is conditioned by its parents. The difficult part in the cost function is the expectation $\left\langle \ln p(X,\theta \mid \mathcal{H}) \right\rangle$ which is computed over the approximation $q(\theta)$ of the posterior pdf. The logarithm splits the product of simple terms into a sum. If each of the simple terms can be computed in constant time, the overall computational complexity is linear.

In general, the computation time is constant if the parents are independent in the posterior pdf approximation $q(\theta)$. This condition is satisfied if the joint distribution of the parents in $q(\theta)$ decouples into the product of the approximate distributions of the parents. That is, each term in $q(\theta)$ depending on the parents depends only on one parent. The independence requirement is violated if any variable receives inputs from a latent variable through multiple computational paths or from two latent variables which are dependent in $q(\theta)$, having a non-factorisable joint distribution there. Figure 1 illustrates the flow of information in the network in these two qualitatively different cases.

Our choice for $q(\theta)$ is a multivariate Gaussian density with a diagonal covariance matrix. Even this crude approximation is adequate for finding the region where the mass of the actual posterior density is concentrated. The mean values of the components of the Gaussian approximation provide reasonably good point estimates of the corresponding parameters or variables, and the respective

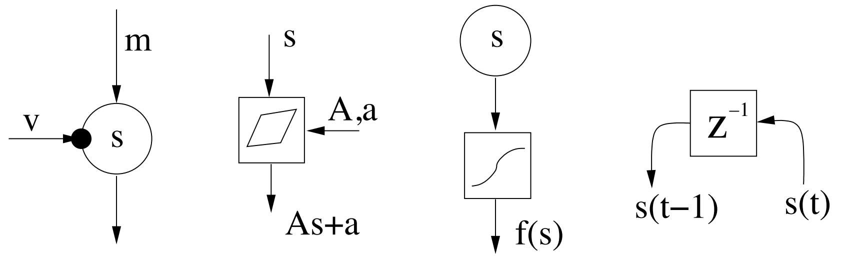


Figure 2: *First subfigure from the left:* The circle represents a Gaussian node corresponding to the latent variable s conditioned by mean m and variance $\exp(-v)$. *Second subfigure:* Addition and multiplication nodes are used to form an affine mapping from s to $As+a$. *Third subfigure:* A nonlinearity f is applied immediately after a Gaussian variable. *The rightmost subfigure:* Delay operator delays a time-dependent signal by one time unit.

variances measure the reliability of these estimates. However, occasionally the diagonal Gaussian approximation can be too crude. This problem has been considered in context with independent component analysis in Ilin and Valpola (2003), giving means to remedy the situation.

Taking into account posterior dependencies makes the posterior pdf approximation $q(\theta)$ more accurate, but also usually increases the computational load significantly. We have earlier considered networks with multiple computational paths in several papers, for example (Lappalainen and Honkela, 2000; Valpola et al., 2002; Valpola and Karhunen, 2002; Valpola et al., 2003b). The computational load of variational Bayesian learning then becomes roughly quadratically proportional to the number of unknown variables in the MLP network model used in Lappalainen and Honkela (2000), Valpola et al. (2003b), and Honkela and Valpola (2005).

The building blocks (nodes) introduced in this paper together with associated structural constraints provide effective means for combating the drawbacks mentioned above. Using them, updating at each node takes place locally with no multiple paths. As a result, the computational load scales linearly with the number of estimated quantities. The cost function and the learning formulas for the unknown quantities to be estimated can be evaluated automatically once a specific model has been selected, that is, after the connections between the blocks used in the model have been fixed. This is a very important advantage of the proposed block approach.

3. Node Types

In this section, we present different types of nodes that can be easily combined together. Variational Bayesian inference algorithm for the nodes is then discussed in Section 4.

In general, the building blocks can be divided into variable nodes, computation nodes, and constants. Each variable node corresponds to a random variable, and it can be either observed or hidden. In this paper we present only one type of variable node, the Gaussian node, but others can be used in the same framework. The computation nodes are the addition node, the multiplication node, a nonlinearity, and the delay node.

In the following, we shall refer to the inputs and outputs of the nodes. For a variable node, its inputs are the parameters of the conditional distribution of the variable represented by that node, and its output is the value of the variable. For computation nodes, the output is a fixed function of the inputs. The symbols used for various nodes are shown in Figure 2. Addition and multiplication nodes are not included, since they are typically combined to represent the effect of a linear transformation, which has a symbol of its own. An output signal of a node can be used as input by zero or more nodes that are called the children of that node. Constants are the only nodes that do not have inputs. The output is a fixed value determined at creation of the node.

Nodes are often structured in vectors or matrices. Assume for example that we have a data matrix $\mathbf{X} = [\mathbf{x}(1), \mathbf{x}(2), \dots, \mathbf{x}(T)]$, where $t = 1, 2, \dots T$ is called the time index of an n-dimensional observation vector. Note that t does not have to correspond to time in the real world, it is just the index of the data case. In the implementation, the nodes are either vectors so that the values indexed by t (observations and latent variables) or scalars so that the values are constants w.r.t. t (parameters). The data $\mathbf{X}$ would be represented with n vector nodes. A weight matrix $\mathbf{A}$ has to be represented with scalar nodes only. A scalar node can be a parent of a vector node, but not a child of a vector node in this sense. In figures and formulae, we use vectors and matrices more freely.

3.1 Gaussian Node

The Gaussian node is a variable node and the basic element in building hierarchical models. Figure 2 (leftmost subfigure) shows the schematic diagram of the Gaussian node. Its output is the value of a Gaussian random variable s, which is conditioned by the inputs m and v. Denote generally by $\mathcal{N}(x; m_x, \sigma_x^2)$ the probability density function of a Gaussian random variable x having the mean m_x and variance σ_x^2. Then the conditional probability function (cpf) of the variable s is $p(s \mid m, v) = \mathcal{N}(s; m, \exp(-v))$. As a generative model, the Gaussian node takes its mean input m and adds to it Gaussian noise (or innovation) with variance $\exp(-v)$.

Variables can be latent or observed. Observing a variable means fixing its output s to the value in the data. Section 4 is devoted to inferring the distribution over the latent variables given the observed variables. Inferring the distribution over variables that are independent of t is also called learning.

3.2 Computation Nodes

The addition and multiplication nodes are used for summing and multiplying variables. These standard mathematical operations are typically used to construct linear mappings between the variables. This task is automated in the software, but in general, the nodes can be connected in other ways, too. An addition node that has n inputs denoted by $s_1, s_2, \dots, s_n$, gives the sum of its inputs as the output $\sum_{i=1}^n s_i$. Similarly, the output of a multiplication node is the product of its inputs $\prod_{i=1}^n s_i$.

A nonlinear computation node can be used for constructing nonlinear mappings between the variable nodes. The nonlinearity

$$f(s) = \exp(-s^2) \tag{7}$$

is chosen because the required expectations can be solved analytically for it. Another implemented nonlinearity for which the computations can be carried out analytically is the cut function $g(s) = \max(s, 0)$. Other possible nonlinearities are discussed in Section 4.3.

3.3 Delay Node

The delay operation can be used to model dynamics. The node operates on time-dependent signals. It transforms the inputs $s(1), s(2), \ldots, s(T)$ into outputs $s_0, s(1), s(2), \ldots, s(T-1)$ where s_0 is a scalar parameter that provides a starting distribution for the dynamical process. The symbol z^{-1} in the rightmost subfigure of Fig. 2 illustrating the delay node is the standard notation for the unit delay in signal processing and temporal neural networks (Haykin, 1998). Models containing the delay node are called dynamic, and the other models are called static.

4. Variational Bayesian Inference in Bayes Blocks

In this section we give equations needed for computation with the nodes introduced in Section 3. Generally speaking, each node propagates to the forward direction a distribution of its output given its inputs. In the backward direction, the dependency of the cost function (4) of the children on the output of their parent is propagated. These two potentials are combined to form the posterior distribution of each variable. There is a direct analogy to Bayes rule (1): the prior (forward) and the likelihood (backward) are combined to form the posterior distribution. We will show later on that the potentials in the two directions are fully determined by a few values, which consist of certain expectations over the distribution in the forward direction, and of gradients of the cost function w.r.t. the same expectations in the backward direction.

In the following, we discuss in more detail the properties of each node. Note that the delay node does not actually process the signals, it just rewires them. Therefore no formulas are needed for its associated the expectations and gradients.

4.1 Gaussian Node

Recall the Gaussian node in Section 3.1. The variance is parameterised using the exponential function as $\exp(-v)$. This is because then the mean $\langle v \rangle$ and expected exponential $\langle \exp v \rangle$ of the input v suffice for evaluating the cost function, as will be shown shortly. Consequently the cost function can be minimised using the gradients with respect to these expectations. The gradients are computed backwards from the children nodes, but otherwise our learning method differs clearly from standard back-propagation (e.g., Haykin, 1998).

Another important reason for using the parameterisation $\exp(-v)$ for the prior variance of a Gaussian random variable s is that the posterior distribution of s then becomes approximately Gaussian, provided that the prior mean m of s is Gaussian, too (see for example Section 7.1 or Lappalainen and Miskin, 2000). The conjugate prior distribution of the inverse of the prior variance of a Gaussian random variable is the gamma distribution (Gelman et al., 1995).

$$p(x \mid \mu, \gamma) = \mathcal{N}\left(x \mid \mu, \gamma^{-1}\right),$$
$$p(\gamma \mid a, b) = \mathrm{Gamma}(\gamma \mid a, b) = \frac{b^a}{\Gamma(a)} \gamma^{a-1} \exp(-b\gamma).$$

Using such gamma prior pdf causes the posterior distribution to be gamma, too, which is mathematically convenient. However, the conjugate prior pdf of the second parameter b of the gamma distribution is something quite intractable. Hence gamma distribution is not suitable for developing hierarchical variance models. The logarithm of a gamma distributed variable is approximately Gaussian distributed (Gelman et al., 1995), justifying the adopted parameterisation $\exp(-v)$. However, it should be noted that both the gamma and $\exp(-v)$ distributions are used as prior pdfs mainly because they make the estimation of the posterior pdf mathematically tractable (Lappalainen and Miskin, 2000); one cannot claim that either of these choices would be correct.

4.1.1 COST FUNCTION

Recall now that we are approximating the joint posterior pdf of the random variables s, m, and v in a maximally factorial manner. It then decouples into the product of the individual distributions: $q(s,m,v) = q(s)q(m)q(v)$. Hence s, m, and v are assumed to be statistically independent a posteriori. The posterior approximation $q(s)$ of the Gaussian variable s is defined to be Gaussian with mean $\bar{s}$ and variance $\tilde{s}$: $q(s) = \mathcal{N}(s;\bar{s},\tilde{s})$. Using these, the part $\mathcal{C}_p$ of the Kullback-Leibler cost function arising from the data, defined in Eq. (6), can be computed in closed form. For the Gaussian node of Figure 2, the cost becomes

$$\begin{aligned}\mathcal{C}_{s,p} &= -\langle \ln p(s|m,v)\rangle \\ &= \frac{1}{2}\left\{ \langle \exp v\rangle \left[(\bar{s}-\langle m\rangle)^2 + \mathrm{Var}\{m\} + \tilde{s}\right] - \langle v\rangle + \ln 2\pi\right\}\end{aligned} \tag{8}$$

The derivation is presented in Appendix B of Valpola and Karhunen (2002) using slightly different notation. For the observed variables, this is the only term arising from them to the cost function $\mathcal{C}_{\mathrm{KL}}$.

However, latent variables contribute to the cost function $\mathcal{C}_{\mathrm{KL}}$ also with the part $\mathcal{C}_q$ defined in Eq. (5), resulting from the expectation $\langle \ln q(s)\rangle$. This term is

$$\mathcal{C}_{s,q} = \int_s q(s)\ln q(s) ds = -\frac{1}{2}[\ln(2\pi\tilde{s}) + 1]$$

which is the negative entropy of Gaussian variable with variance $\tilde{s}$. The parameters defining the approximation $q(s)$ of the posterior distribution of s, namely its mean $\bar{s}$ and variance $\tilde{s}$, are to be optimised during learning.

The output of a latent Gaussian node trivially provides the mean and the variance: $\langle s\rangle = \bar{s}$ and $\mathrm{Var}\{s\} = \tilde{s}$. The expected exponential can be easily shown to be (Lappalainen and Miskin, 2000; Valpola and Karhunen, 2002)

$$\langle \exp s\rangle = \exp(\bar{s} + \tilde{s}/2). \tag{9}$$

The outputs of the nodes corresponding to the observations are known scalar values instead of distributions. Therefore for these nodes $\langle s\rangle = s$, $\mathrm{Var}\{s\} = 0$, and $\langle \exp s\rangle = \exp s$. An important conclusion of the considerations presented this far is that the cost function of a Gaussian node can be computed analytically in a closed form. This requires that the posterior approximation is Gaussian and that the mean $\langle m\rangle$ and the variance $\mathrm{Var}\{m\}$ of the mean input m as well as the mean $\langle v\rangle$ and the expected exponential $\langle \exp v\rangle$ of the variance input v can be computed. To summarise, we have shown that Gaussian nodes can be connected together and their costs can be evaluated analytically.

We will later on use derivatives of the cost function with respect to some expectations of its mean and variance parents m and v as messages from children to parents. They are derived directly from Eq. (8), taking the form

$$\frac{\partial \mathcal{C}_{s,p}}{\partial \langle m \rangle} = \langle \exp v \rangle (\langle m \rangle - \overline{s}), \tag{10}$$

$$\frac{\partial \mathcal{C}_{s,p}}{\partial \mathrm{Var}\{m\}} = \frac{\langle \exp v \rangle}{2}, \tag{11}$$

$$\frac{\partial \mathcal{C}_{s,p}}{\partial \langle v \rangle} = -\frac{1}{2}, \tag{12}$$

$$\frac{\partial \mathcal{C}_{s,p}}{\partial \langle \exp v \rangle} = \frac{(\overline{s} - \langle m \rangle)^2 + \mathrm{Var}\{m\} + \widetilde{s}}{2}. \tag{13}$$

4.1.2 UPDATING THE POSTERIOR DISTRIBUTION

The posterior distribution $q(s)$ of a latent Gaussian node can be updated as follows.

1. The distribution $q(s)$ affects the terms of the cost function $\mathcal{C}_s$ arising from the variable s itself, namely $\mathcal{C}_{s,p}$ and $\mathcal{C}_{s,q}$, as well as the $\mathcal{C}_p$ terms of the children of s, denoted by $\mathcal{C}_{\mathrm{ch}(s),p}$. The gradients of the cost $\mathcal{C}_{\mathrm{ch}(s),p}$ with respect to $\langle s \rangle$, $\mathrm{Var}\{s\}$, and $\langle \exp s \rangle$ are computed according to Equations (10–13).

2. The terms in $\mathcal{C}_p$ which depend on $\overline{s}$ and $\widetilde{s}$ can be shown (see Appendix B.2) to be of the form 2

$$\mathcal{C}_p = \mathcal{C}_{s,p} + \mathcal{C}_{\mathrm{ch}(s),p} = M\overline{s} + V[(\overline{s} - \overline{s}_{\mathrm{current}})^2 + \widetilde{s}] + E\langle \exp s \rangle, \tag{14}$$

   where

$$M = \frac{\partial \mathcal{C}_p}{\partial \overline{s}}, \qquad V = \frac{\partial \mathcal{C}_p}{\partial \widetilde{s}}, \quad \text{and } E = \frac{\partial \mathcal{C}_p}{\partial \langle \exp s \rangle}.$$

3. The minimum of $\mathcal{C}_s = \mathcal{C}_{s,p} + \mathcal{C}_{s,q} + \mathcal{C}_{\mathrm{ch}(s),p}$ is solved. This can be done analytically if $E = 0$, corresponding to the case of so-called free-form solution (see Lappalainen and Miskin, 2000, for details):

$$\overline{s}_{\mathrm{opt}} = \overline{s}_{\mathrm{current}} - \frac{M}{2V}, \qquad \widetilde{s}_{\mathrm{opt}} = \frac{1}{2V}.$$

   Otherwise the minimum is obtained iteratively. Iterative minimisation can be carried out efficiently using Newton's method for the posterior mean $\overline{s}$ and a fixed-point iteration for the posterior variance $\widetilde{s}$. The minimisation procedure is discussed in more detail in Appendix A.

4.2 Addition and Multiplication Nodes

Consider first the addition node. The mean, variance and expected exponential of the output of the addition node can be evaluated in a straightforward way. Assuming that the inputs s_i are statistically

2. Note that constants are dropped out since they do not depend on $\overline{s}$ or $\widetilde{s}$.

independent, these expectations are respectively given by

$$\left\langle \sum_{i=1}^{n} s_i \right\rangle = \sum_{i=1}^{n} \langle s_i \rangle , \tag{15}$$

$$\operatorname{Var}\left\{ \sum_{i=1}^{n} s_i \right\} = \sum_{i=1}^{n} \operatorname{Var}\{s_i\} , \tag{16}$$

$$\left\langle \exp\left(\sum_{i=1}^{n} s_i \right) \right\rangle = \prod_{i=1}^{n} \langle \exp s_i \rangle . \tag{17}$$

The proof has been given in Appendix B.1.

Consider then the multiplication node. Assuming independence between the inputs s_i, the mean and the variance of the output take the form (see Appendix B.1)

$$\left\langle \prod_{i=1}^{n} s_i \right\rangle = \prod_{i=1}^{n} \langle s_i \rangle , \tag{18}$$

$$\operatorname{Var}\left\{ \prod_{i=1}^{n} s_i \right\} = \prod_{i=1}^{n} \left[\langle s_i \rangle^2 + \operatorname{Var}\{s_i\} \right] - \prod_{i=1}^{n} \langle s_i \rangle^2 . \tag{19}$$

For the multiplication node the expected exponential cannot be evaluated without knowing the exact distribution of the inputs.

The formulas (15)–(19) are given for n inputs because of generality, but in practice we have carried out the needed calculations pairwise. When using the general formula (19), the variance might otherwise occasionally take a small negative value due to minor imprecisions appearing in the computations. This problem does not arise in pairwise computations. Now, the propagation in the forward direction is covered.

The form of the cost function propagating from children to parents is assumed to be of the form (14). This is true even in the case, where there are addition and multiplication nodes in between (see Appendix B.2 for proof). Therefore only the gradients of the cost function with respect to the different expectations need to be propagated backwards to identify the whole cost function w.r.t. the parent. The required formulas are obtained in a straightforward manner from Eqs. (15)–(19). The gradients for the addition node are:

$$\frac{\partial C}{\partial \langle s_1 \rangle} = \frac{\partial C}{\partial \langle s_1 + s_2 \rangle}, \tag{20}$$

$$\frac{\partial C}{\partial \operatorname{Var}\{s_1\}} = \frac{\partial C}{\partial \operatorname{Var}\{s_1 + s_2\}}, \tag{21}$$

$$\frac{\partial C}{\partial \langle \exp s_1 \rangle} = \langle \exp s_2 \rangle \frac{\partial C}{\partial \langle \exp(s_1 + s_2) \rangle}. \tag{22}$$

For the multiplication node, they become

$$\frac{\partial C}{\partial \langle s_1 \rangle} = \langle s_2 \rangle \frac{\partial C}{\partial \langle s_1 s_2 \rangle} + 2\operatorname{Var}\{s_2\} \frac{\partial C}{\partial \operatorname{Var}\{s_1 s_2\}} \langle s_1 \rangle , \tag{23}$$

$$\frac{\partial C}{\partial \operatorname{Var}\{s_1\}} = \left(\langle s_2 \rangle^2 + \operatorname{Var}\{s_2\} \right) \frac{\partial C}{\partial \operatorname{Var}\{s_1 s_2\}}. \tag{24}$$

As a conclusion, addition and multiplication nodes can be added between the Gaussian nodes whose costs still retain the form (14). Proofs can be found in Appendices B.1 and B.2.

4.3 Nonlinearity Node

A serious problem arising here is that for most nonlinear functions it is impossible to compute the required expectations analytically. Here we describe a particular nonlinearity in detail and discuss the options for extending to other nonlinearities, for which the implementation is underway.

Ghahramani and Roweis have shown (1999) that for the nonlinear function $f(s) = \exp(-s^2)$ in Eq. (7), the mean and variance have analytical expressions, to be presented shortly, provided that it has Gaussian input. In our graphical network structures this condition is fulfilled if we require that the nonlinearity must be inserted immediately after a Gaussian node. The same type of exponential function (7) is frequently used in standard radial-basis function networks (Bishop, 1995; Haykin, 1998; Ghahramani and Roweis, 1999), but in a different manner. There the exponential function depends on the Euclidean distance from a centre point, while in our case it depends on the input variable s directly.

The first and second moments of the function (7) with respect to the distribution $q(s)$ are (Ghahramani and Roweis, 1999)

$$\langle f(s)\rangle = \exp\left(-\frac{\overline{s}^2}{2\widetilde{s}+1}\right)(2\widetilde{s}+1)^{-\frac{1}{2}}, \tag{25}$$

$$\left\langle [f(s)]^2\right\rangle = \exp\left(-\frac{2\overline{s}^2}{4\widetilde{s}+1}\right)(4\widetilde{s}+1)^{-\frac{1}{2}}. \tag{26}$$

The formula (25) provides directly the mean $\langle f(s)\rangle$, and the variance is obtained from (25) and (26) by applying the familiar formula $\mathrm{Var}\{f(s)\} = \left\langle [f(s)]^2\right\rangle - \langle f(s)\rangle^2$. The expected exponential $\langle \exp f(s)\rangle$ cannot be evaluated analytically, which limits somewhat the use of the nonlinear node.

The updating of the nonlinear node following directly a Gaussian node takes place similarly as the updating of a plain Gaussian node. The gradients of $\mathcal{C}_p$ with respect to $\langle f(s)\rangle$ and $\mathrm{Var}\{f(s)\}$ are evaluated assuming that they arise from a quadratic term. This assumption holds since the nonlinearity can only propagate to the mean of Gaussian nodes. The update formulas are given in Appendix C.

Another possibility is to use as the nonlinearity the error function $f(s) = \int_{-\infty}^{s} \exp(-r^2)dr$, because its mean can be evaluated analytically and variance approximated from above (Frey and Hinton, 1999). Increasing the variance increases the value of the cost function, too, and hence it suffices to minimise the upper bound of the cost function for finding a good solution. Frey and Hinton (1999) apply the error function in MLP (multilayer perceptron) networks (Bishop, 1995; Haykin, 1998) but in a manner different from ours.

Finally, Murphy (1999) has applied the hyperbolic tangent function $f(s) = \tanh(s)$, approximating it iteratively with a Gaussian. Honkela and Valpola (2005) approximate the same sigmoidal function with a Gauss-Hermite quadrature. This alternative could be considered here, too. A problem with it is, however, that the cost function (mean and variance) cannot be computed analytically.

4.4 Other Possible Nodes

One of the authors has recently implemented two new variable nodes (Harva, 2004; Harva et al., 2005) into the Bayes Blocks software library. They are the mixture-of-Gaussians (MoG) node and

Node type	$\langle\cdot\rangle$	$\mathrm{Var}\{\cdot\}$	$\langle\exp\cdot\rangle$
Gaussian node	$\overline{s}$	$\widetilde{s}$	(9)
Addition node	(15)	(16)	(17)
Multiplication node	(18)	(19)	-
Nonlinearity	(25)	(25),(26)	-
Constant	c	0	$\exp c$

Table 1: The forward messages or expectations that are provided by the output of different types of nodes. The numbers in parentheses refer to defining equations. The multiplication and nonlinearity cannot provide the expected exponential.

the rectified Gaussian node. MoG can be used to model any sufficiently well behaving distribution (Bishop, 1995). In the independent factor analysis (IFA) method introduced in Attias (1999), a MoG distribution was used for the sources, resulting in a probabilistic version of independent component analysis (ICA) (Hyvärinen et al., 2001).

The second new node type, the rectified Gaussian variable, was introduced in Miskin and MacKay (2001). By omitting negative values and retaining only positive ones of a variable which is originally Gaussian distributed, this block allows modelling of variables having positive values only. Such variables are commonplace for example in digital image processing, where the picture elements (pixels) have always non-negative values. The cost functions and update rules of the MoG and rectified Gaussian node have been derived by Harva (2004). We postpone a more detailed discussion of these nodes to forthcoming papers to keep the length of this paper reasonable.

In the early conference paper (Valpola et al., 2001) where we introduced the blocks for the first time, two more blocks were proposed for handling discrete models and variables. One of them is a switch, which picks up its k-th continuous valued input signal as its output signal. The other one is a discrete variable k, which has a soft-max prior derived from the continuous valued input signals c_i of the node. However, we have omitted these two nodes from the present paper, because their performance has not turned out to be adequate. The reason might be that assuming posterior independence of the auxiliary Gaussian variables is too rough and that the cost function in the soft-max case could not be computed exactly. For instance, building a mixture-of-Gaussians from discrete and Gaussian variables with switches is possible, but the construction loses out to a specialised MoG node that makes fewer assumptions. Raiko (2005) uses the soft-max node without switches.

Action and utility nodes (Pearl, 1988; Murphy, 2001) would extend the library into decision theory and control. In addition to the messages about the variational Bayesian cost function, the network would propagate messages about utility. Raiko and Tornio (2005) describe such a system in a slightly different framework.

5. Combining the Nodes

The expectations provided by the outputs and required by the inputs of the different nodes are summarised in Tables 1 and 2, respectively. One can see that the variance input of a Gaussian node requires the expected exponential of the incoming signal. However, it cannot be computed for the nonlinear and multiplication nodes. Hence all the nodes cannot be combined freely.

Input type	$\frac{\partial \mathcal{C}}{\partial \langle \cdot \rangle}$	$\frac{\partial \mathcal{C}}{\partial \mathrm{Var}\{\cdot\}}$	$\frac{\partial \mathcal{C}}{\partial \langle \exp \cdot \rangle}$
Mean of a Gaussian node	(10)	(11)	0
Variance of a Gaussian node	(12)	0	(13)
Addendum	(20)	(21)	(22)
Factor	(23)	(24)	0

Table 2: The backward messages or the gradients of the cost function w.r.t. certain expectations. The numbers in parentheses refer to defining equations. The gradients of the Gaussian node are derived from Eq. (8). The Gaussian node requires the corresponding expectations from its inputs, that is, $\langle m \rangle$, $\mathrm{Var}\{m\}$, $\langle v \rangle$, and $\langle \exp v \rangle$. Addition and multiplication nodes require the same type of input expectations that they are required to provide as output. Communication of a nonlinearity with its Gaussian parent node is described in Appendix C.

When connecting the nodes, the following restrictions must be taken into account:

1. In general, the network has to be a directed acyclic graph (DAG). The delay nodes are an exception because the past values of any node can be the parents of any other nodes. This violation is not a real one in the sense that if the structure were unfolded in time, the resulting network would again be a DAG.

2. The nonlinearity must always be placed immediately after a Gaussian node. This is because the output expectations, Equations (25) and (26), can be computed only for Gaussian inputs. The nonlinearity also breaks the general form of the likelihood (14). This is handled by using special update rules for the Gaussian followed by a nonlinearity (Appendix C).

3. The outputs of multiplication and nonlinear nodes cannot be used as variance inputs for the Gaussian node. This is because the expected exponential cannot be evaluated for them. These restrictions are evident from Tables 1 and 2.

4. There should be only one computational path from a latent variable to a variable. A computational path is a path that consists of computation nodes only. Otherwise, the independency assumptions used in Equations (8) and (16)–(19) are violated and variational Bayesian learning becomes more complicated (recall Figure 1).

Figure 3 shows examples of model structures that break each of the restrictions in turn. The software package includes a function that checks whether a given model structure is correct or not. Note that the network may contain loops, that is, the underlying undirected network can be cyclic. Note also that the second, third, and fourth restrictions can be circumvented by inserting mediating Gaussian nodes. A mediating Gaussian node that is used as the variance input of another variable, is called the variance source and it is discussed in the following.

5.1 Nonstationary Variance

In most currently used models, only the means of Gaussian nodes have hierarchical or dynamical models. In many real-world situations the variance is not a constant either but it is more difficult to

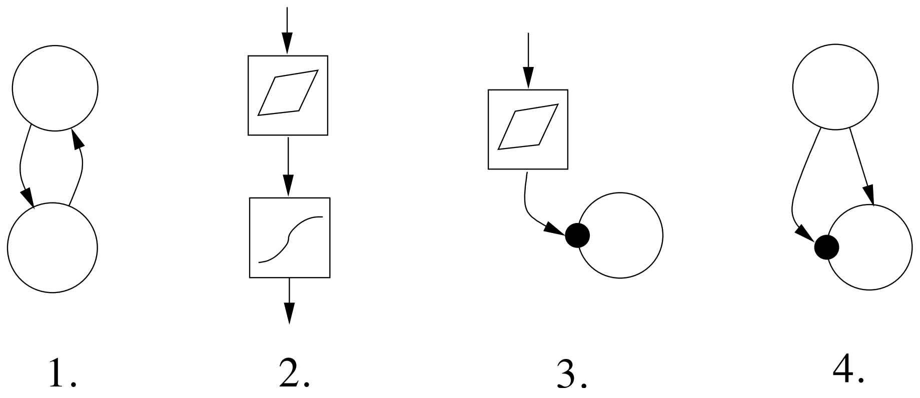


Figure 3: Examples of illegal model structures violating each of the restriction described in the text in turn.

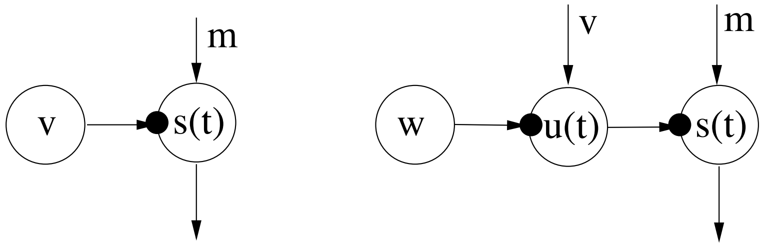


Figure 4: *Left:* The Gaussian variable $s(t)$ has a a constant variance $\exp(-v)$ and mean m. *Right:* A variance source is added for providing a non-constant variance input $u(t)$ to the output (source) signal $s(t)$. The variance source $u(t)$ has a prior mean v and prior variance $\exp(-w)$.

model it. For modelling the variance, too, we use the variance source (Valpola et al., 2004) depicted schematically in Figure 4. Variance source is a regular Gaussian node whose output $u(t)$ is used as the input variance of another Gaussian node. Variance source can convert prediction of the mean into prediction of the variance, allowing to build hierarchical or dynamical models for the variance.

The output $s(t)$ of a Gaussian node to which the variance source is attached (see the right subfigure of Fig. 4) has in general a super-Gaussian distribution. Such a distribution is typically characterised by long tails and a high peak, and it is formally defined as having a positive value of kurtosis (see Hyvärinen et al. (2001) for a detailed discussion). This property has been proved by Parra et al. (2001) who showed that a nonstationary variance (amplitude) always increases the kurtosis. The output signal $s(t)$ of the stationary Gaussian variance source depicted in the left subfigure of Fig. 4 is naturally Gaussian distributed with zero kurtosis. The variance source is useful in modelling natural signals such as speech and images which are typically super-Gaussian, and also in modelling outliers in the observations.

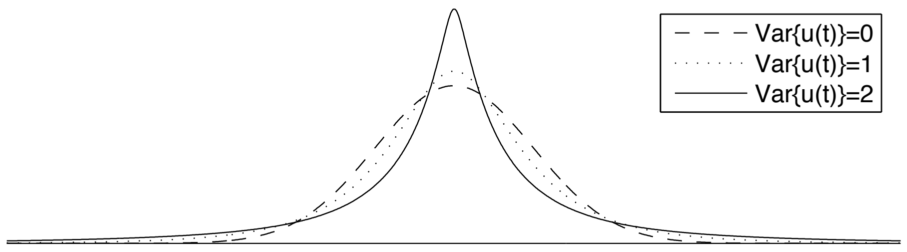


Figure 5: The distribution of $s(t)$ is plotted when $s(t) \sim \mathcal{N}(0, \exp[-u(t)])$ and $u(t) \sim \mathcal{N}(0, \cdot)$. Note that when $\mathrm{Var}\{u(t)\} = 0$, the distribution of $s(t)$ is Gaussian. This corresponds to the right subfigure of Fig. 4 when $m = v = 0$ and $\exp(-w) = 0, 1, 2$.

5.2 Linear Independent Factor Analysis

In many instances there exist several nodes which have quite similar role in the chosen structure. Assuming that i^{th} such node corresponds to a scalar variable y_i, it is convenient to use the vector $\mathbf{y} = (y_1, y_2, \ldots, y_n)^T$ to jointly denote all the corresponding scalar variables $y_1, y_2, \ldots, y_n$. This notation is used in Figures 6 and 7 later on. Hence we represent the scalar source nodes corresponding to the variables $s_i(t)$ using the source vector $\mathbf{s}(t)$, and the scalar nodes corresponding to the observations $x_i(t)$ using the observation vector $\mathbf{x}(t)$.

The addition and multiplication nodes can be used for building an affine transformation

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{a} + \mathbf{n}_x(t) \tag{27}$$

from the Gaussian source nodes $\mathbf{s}(t)$ to the Gaussian observation nodes $\mathbf{x}(t)$. The vector $\mathbf{a}$ denotes the bias and vector $\mathbf{n}_x(t)$ denotes the zero-mean Gaussian noise in the Gaussian node $\mathbf{x}(t)$. This model corresponds to standard linear factor analysis (FA) assuming that the sources $s_i(t)$ are mutually uncorrelated; see for example, Hyvärinen et al. (2001).

If instead of Gaussianity it is assumed that each source $s_i(t)$ has some non-Gaussian prior, the model (27) describes linear independent factor analysis (IFA). Linear IFA was introduced by Attias (1999), who used variational Bayesian learning for estimating the model except for some parts which he estimated using the expectation-maximisation (EM) algorithm. Attias used a mixture-of-Gaussians source model, but another option is to use the variance source to achieve a super-Gaussian source model. Figure 6 depicts the model structures for linear factor analysis and independent factor analysis.

5.3 A Hierarchical Variance Model

Figure 7 (right subfigure) presents a hierarchical model for the variance, and also shows how it can be constructed by first learning simpler structures shown in the left and middle subfigures of Fig. 7. This is necessary, because learning a hierarchical model having different types of nodes from scratch in a completely unsupervised manner would be too demanding a task, ending quite probably into an unsatisfactory local minimum.

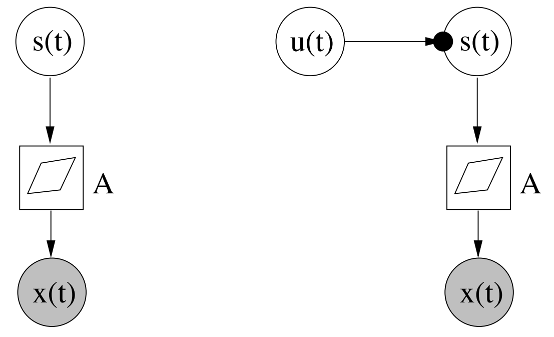


Figure 6: Model structures for linear factor analysis (FA) (left) and independent factor analysis (IFA) (right).

The final rightmost variance model in Fig. 7 is somewhat involved in that it contains both nonlinearities and hierarchical modelling of variances. Before going into its mathematical details and into the two simpler models in Fig. 7, we point out that we have considered in our earlier papers related but simpler block models. In Valpola et al. (2003c), a hierarchical nonlinear model for the data $\mathbf{x}(t)$ is discussed without modelling the variance. Such a model can be applied for example to nonlinear ICA or blind source separation. Experimental results (Valpola et al., 2003c) show that this block model performs adequately in the nonlinear BSS problem, even though the results are slightly poorer than for our earlier computationally more demanding model (Lappalainen and Honkela, 2000; Valpola et al., 2003b; Honkela and Valpola, 2005) with multiple computational paths.

In another paper (Valpola et al., 2004), we have considered hierarchical modelling of variance using the block approach without nonlinearities. Experimental results on biomedical MEG (magnetoencephalography) data demonstrate the usefulness of hierarchical modelling of variances and existence of variance sources in real-world data.

Learning starts from the simple structure shown in the left subfigure of Fig. 7. There a variance source is attached to each Gaussian observation node. The nodes represent vectors, with $\mathbf{u}_1(t)$ being the output vector of the variance source and $\mathbf{x}(t)$ the t^{th} observation (data) vector. The vectors $\mathbf{u}_1(t)$ and $\mathbf{x}(t)$ have the same dimension, and each component of the variance vector $\mathbf{u}_1(t)$ models the variance of the respective component of the observation vector $\mathbf{x}(t)$.

Mathematically, this simple first model obeys the equations

$$\mathbf{x}(t) = \mathbf{a}_1 + \mathbf{n}_x(t), \tag{28}$$

$$\mathbf{u}_1(t) = \mathbf{b}_1 + \mathbf{n}_{u_1}(t). \tag{29}$$

Here the vectors $\mathbf{a}_1$ and $\mathbf{b}_1$ denote the constant means (bias terms) of the data vector $\mathbf{x}(t)$ and the variance variable vector $\mathbf{u}_1(t)$, respectively. The additive "noise" vector $\mathbf{n}_x(t)$ determines the variances of the components of $\mathbf{x}(t)$. It has a Gaussian distribution with a zero mean and variance $\exp[-\mathbf{u}_1(t)]$:

$$\mathbf{n}_x(t) \sim \mathcal{N}(\mathbf{0}, \exp[-\mathbf{u}_1(t)]). \tag{30}$$

More precisely, the shorthand notation $\mathcal{N}(\mathbf{0}, \exp[-\mathbf{u}_1(t)])$ means that each component of $\mathbf{n}_x(t)$ is Gaussian distributed with a zero mean and variance defined by the respective component of the vector $\exp[-\mathbf{u}_1(t)]$. The exponential function $\exp(\cdot)$ is applied separately to each component of the

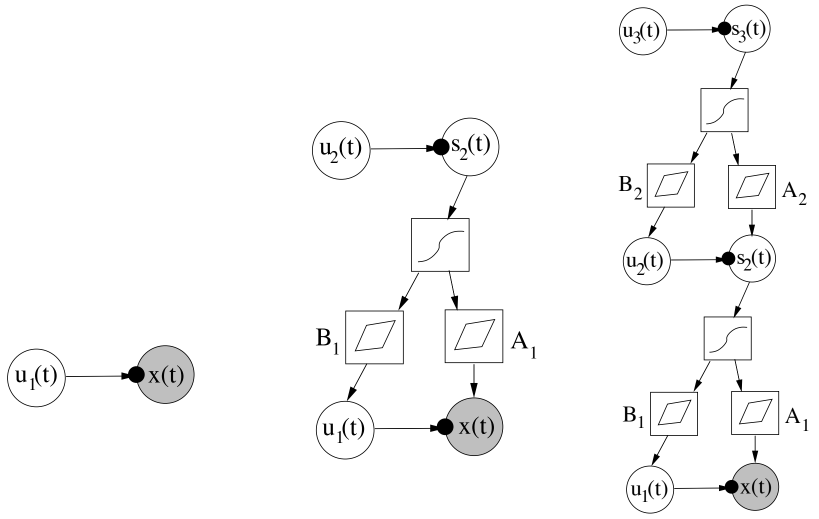


Figure 7: Construction of a hierarchical variance model in stages from simpler models. *Left:* In the beginning, a variance source is attached to each Gaussian observation node. The nodes represent vectors. *Middle:* A layer of sources with variance sources attached to them is added. They layers are connected through a nonlinearity and an affine mapping. *Right:* Another layer is added on the top to form the final hierarchical variance model.

vector $-\mathbf{u}_1(t)$. Similarly,

$$\mathbf{n}_{u_1}(t) \sim \mathcal{N}(\mathbf{0}, \exp[-\mathbf{v}_1]) \tag{31}$$

where the components of the vector $\mathbf{v}_1$ define the variances of the zero mean Gaussian variables $\mathbf{n}_{u_1}(t)$.

Consider then the intermediate model shown in the middle subfigure of Fig. 7. In this second learning stage, a layer of sources with variance sources attached to them is added. These sources are represented by the source vector $\mathbf{s}_2(t)$, and their variances are given by the respective components of the variance vector $\mathbf{u}_2(t)$ quite similarly as in the left subfigure. The (vector) node between the source vector $\mathbf{s}_2(t)$ and the variance vector $\mathbf{u}_1(t)$ represents an affine transformation with a transformation matrix $\mathbf{A}_1$ including a bias term. Hence the prior mean inputted to the Gaussian variance source having the output $\mathbf{u}_1(t)$ is of the form $\mathbf{B}_1\mathbf{f}(\mathbf{s}_2(t)) + \mathbf{b}_1$, where $\mathbf{b}_1$ is the bias vector, and $\mathbf{f}(\cdot)$ is a vector of componentwise nonlinear functions (7). Quite similarly, the vector node between $\mathbf{s}_2(t)$ and the observation vector $\mathbf{x}(t)$ yields as its output the affine transformation $\mathbf{A}_1\mathbf{f}(\mathbf{s}_2(t)) + \mathbf{a}_1$, where $\mathbf{a}_1$ is a bias vector. This in turn provides the input prior mean to the Gaussian node modelling the observation vector $\mathbf{x}(t)$.

The mathematical equations corresponding to the model represented graphically in the middle subfigure of Fig. 7 are:

$$\mathbf{x}(t) = \mathbf{A}_1\mathbf{f}(\mathbf{s}_2(t)) + \mathbf{a}_1 + \mathbf{n}_x(t), \tag{32}$$

$$\mathbf{u}_1(t) = \mathbf{B}_1\mathbf{f}(\mathbf{s}_2(t)) + \mathbf{b}_1 + \mathbf{n}_{u_1}(t), \tag{33}$$

$$\mathbf{s}_2(t) = \mathbf{a}_2 + \mathbf{n}_{s_2}(t), \tag{34}$$

$$\mathbf{u}_2(t) = \mathbf{b}_2 + \mathbf{n}_{u_2}(t). \tag{35}$$

Compared with the simplest model (28)–(29), one can observe that the source vector $\mathbf{s}_2(t)$ of the second (upper) layer and the associated variance vector $\mathbf{u}_2(t)$ are of quite similar form, given in Eqs. (34)–(35). The models (32)–(33) of the data vector $\mathbf{x}(t)$ and the associated variance vector $\mathbf{u}_1(t)$ in the first (bottom) layer differ from the simple first model (28)–(29) in that they contain additional terms $\mathbf{A}_1\mathbf{f}(\mathbf{s}_2(t))$ and $\mathbf{B}_1\mathbf{f}(\mathbf{s}_2(t))$, respectively. In these terms, the nonlinear transformation $\mathbf{f}(\mathbf{s}_2(t))$ of the source vector $\mathbf{s}_2(t)$ coming from the upper layer have been multiplied by the linear mixing matrices $\mathbf{A}_1$ and $\mathbf{B}_1$. All the "noise" terms $\mathbf{n}_x(t)$, $\mathbf{n}_{u_1}(t)$, $\mathbf{n}_{s_2}(t)$, and $\mathbf{n}_{u_2}(t)$ in Eqs. (32)–(35) are modelled by similar zero mean Gaussian distributions as in Eqs. (30) and (31).

In the last stage of learning, another layer is added on the top of the network shown in the middle subfigure of Fig. 7. The resulting structure is shown in the right subfigure. The added new layer is quite similar as the layer added in the second stage. The prior variances represented by the vector $\mathbf{u}_3(t)$ model the source vector $\mathbf{s}_3(t)$, which is turn affects via the affine transformation $\mathbf{B}_2\mathbf{f}(\mathbf{s}_3(t)) + \mathbf{b}_2$ to the mean of the mediating variance node $\mathbf{u}_2(t)$. The source vector $\mathbf{s}_3(t)$ provides also the prior mean of the source $\mathbf{s}_2(t)$ via the affine transformation $\mathbf{A}_2\mathbf{f}(\mathbf{s}_3(t)) + \mathbf{a}_2$.

The model equations (32)–(33) for the data vector $\mathbf{x}(t)$ and its associated variance vector $\mathbf{u}_1(t)$ remain the same as in the intermediate model shown graphically in the middle subfigure of Fig. 7. Note that $\mathbf{A}_1$ and $\mathbf{B}_1$ are still updated. The model equations of the second and third layer sources $\mathbf{s}_2(t)$ and $\mathbf{s}_3(t)$ as well as their respective variance vectors $\mathbf{u}_2(t)$ and $\mathbf{u}_3(t)$ in the rightmost subfigure of Fig. 7 are given by

$$\begin{aligned}
\mathbf{s}_2(t) &= \mathbf{A}_2\mathbf{f}(\mathbf{s}_3(t)) + \mathbf{a}_2 + \mathbf{n}_{s_2}(t), \\
\mathbf{u}_2(t) &= \mathbf{B}_2\mathbf{f}(\mathbf{s}_3(t)) + \mathbf{b}_2 + \mathbf{n}_{u_2}(t), \\
\mathbf{s}_3(t) &= \mathbf{a}_3 + \mathbf{n}_{s_3}(t), \\
\mathbf{u}_3(t) &= \mathbf{b}_3 + \mathbf{n}_{u_3}(t).
\end{aligned}$$

Again, the vectors $\mathbf{a}_2, \mathbf{b}_2, \mathbf{a}_3$, and $\mathbf{b}_3$ represent the constant means (biases) in their respective models, and $\mathbf{A}_2$ and $\mathbf{B}_2$ are mixing matrices with matching dimensions. The vectors $\mathbf{n}_{s_2}(t)$, $\mathbf{n}_{u_2}(t)$, $\mathbf{n}_{s_3}(t)$, and $\mathbf{n}_{u_3}(t)$ have similar zero mean Gaussian distributions as in Eqs. (30) and (31).

It should be noted that in the resulting network the number of scalar-valued nodes (size of the layers) can be different for different layers. Additional layers could be appended in the same manner. The final network of the right subfigure in Fig. 7 uses variance nodes in building a hierarchical model for both the means and variances. Without the variance sources the model would correspond to a nonlinear model with latent variables in the hidden layer. As already mentioned, we have considered such a nonlinear hierarchical model in Valpola et al. (2003c). Note that as latent variables of the upper layer are connected to observations through multiple hidden nodes, having computation nodes as hidden nodes would result in multiple computational paths. This type of structure was used in Lappalainen and Honkela (2000), and it has a quadratic computational complexity as opposed to linear one of the networks in Figure 7.

5.4 Linear Dynamic Models for the Sources and Variances

Sometimes it is useful to complement the linear factor analysis model

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{a} + \mathbf{n}_x(t) \tag{36}$$

with a recursive one-step prediction model for the source vector $\mathbf{s}(t)$:

$$\mathbf{s}(t) = \mathbf{B}\mathbf{s}(t-1) + \mathbf{b} + \mathbf{n}_s(t). \tag{37}$$

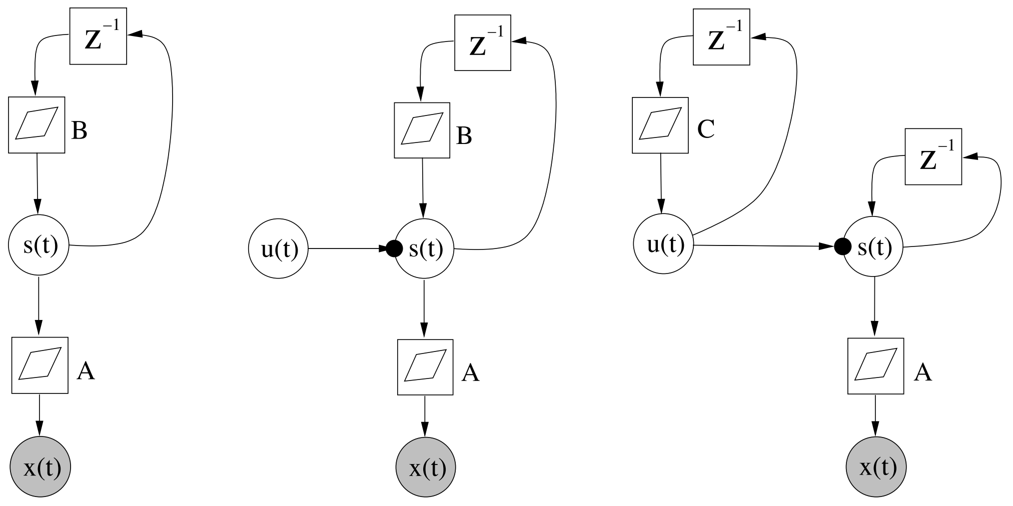


Figure 8: Three model structures. A linear Gaussian state-space model (left); the same model complemented with a super-Gaussian innovation process for the sources (middle); and a dynamic model for the variances of the sources which also have a recurrent dynamic model (right).

The noise term $\mathbf{n}_s(t)$ is called the innovation process. The dynamic model of the type (36), (37) is used for example in Kalman filtering (Haykin, 1998, 2001), but other estimation algorithms can be applied as well (Haykin, 1998). The left subfigure in Fig. 8 depicts the structure arising from Eqs. (36) and (37), built from the blocks.

A straightforward extension is to use variance sources for the sources to make the innovation process super-Gaussian. The variance signal $\mathbf{u}(t)$ characterises the innovation process of $\mathbf{s}(t)$, in effect telling how much the signal differs from the predicted one but not in which direction it is changing. The graphical model of this extension is depicted in the middle subfigure of Fig. 8. The mathematical equations describing this model can be written in a similar manner as for the hierarchical variance models in the previous subsection.

Another extension is to model the variance sources dynamically by using one-step recursive prediction model for them:

$$\mathbf{u}(t) = \mathbf{C}\mathbf{u}(t-1) + \mathbf{c} + \mathbf{n}_u(t).$$

This model is depicted graphically in the rightmost subfigure of Fig. 8. In context with it, we use the simplest possible identity dynamical mapping for $\mathbf{s}(t)$:

$$\mathbf{s}(t) = \mathbf{s}(t-1) + \mathbf{n}_s(t).$$

The latter two models introduced in this subsection will be tested experimentally later on in this paper.

5.5 Hierarchical Priors

It is often desirable that the priors of the parameters should not be too restrictive. A common type of a vague prior is the hierarchical prior (Gelman et al., 1995). For example the priors of the elements

a_{ij} of a mixing matrix $\mathbf{A}$ can be defined via the Gaussian distributions

$$p(a_{ij} \mid v_i^a) = \mathcal{N}(a_{ij}; 0, \exp(-v_i^a)),$$
$$p(v_i^a \mid m^{va}, v^{va}) = \mathcal{N}(v_i^a; m^{va}, \exp(-v^{va}))\,.$$

Finally, the priors of the quantities m^{va} and v^{va} have flat Gaussian distributions $\mathcal{N}(\cdot;0,100)$ (the constants depending on the scale of the data). When going up in the hierarchy, we use the same distribution for each column of a matrix and for each component of a vector. On the top, the number of required constant priors is small. Thus very little information is provided and needed a priori. This kind of hierarchical priors are used in the experiments later on this paper.

6. Learning

Let us now discuss the overall learning procedure, describing also briefly how problems related with learning can be handled.

6.1 Updating of the Network

The nodes of the network communicate with their parents and children by providing certain expectations in the feedforward direction (from parents to children) and gradients of the cost function with respect to the same expectations in the feedback direction (from children to parents). These expectations and gradients are summarised in Tables 1 and 2.

The basic element for updating the network is the update of a single node assuming the rest of the network fixed. For computation nodes this is simple: each time when a child node asks for expectations and they are out of date, the computational node asks from its parents for their expectations and updates its own ones. And vice versa: when parents ask for gradients and they are out of date, the node asks from its children for the gradients and updates its own ones. These updates have analytical formulas given in Section 4.

For a variable node to be updated, the input expectations and output gradients need to be up-to-date. The posterior approximation $q(s)$ can then be adjusted to minimise the cost function as explained in Section 4. The minimisation is either analytical or iterative, depending on the situation. Signals propagating outwards from the node (the output expectations and the input gradients) of a variable node are functions of $q(s)$ and are thus updated in the process. Each update is guaranteed not to increase the cost function.

One sweep of updating means updating each node once. The order in which this is done is not critical for the system to work. It would not be useful to update a variable twice without updating some of its neighbours in between, but that does not happen with any ordering when updates are done in sweeps. We have used an ordering where each variable node is updated only after all of its descendants have been updated. Basically when a variable node is updated, its input gradients and output expectations are labelled as outdated and they are updated only when another node asks for that information.

It is possible to use different measures to improve the learning process. Measures for avoiding local minima are described in the next subsection. Another enhancement can be used for speeding up learning. The basic idea is that after some time, the parameters describing $q(s)$ are changing fairly linearly between consecutive sweeps. Therefore a line search in that direction provides faster learning, as discussed in Honkela (2002); Honkela et al. (2003). We apply this line search only at every tenth sweep for allowing the consecutive updates to become fairly linear again.

Learning a model typically takes thousands of sweeps before convergence. The cost function decreases monotonically after every update. Typically this decrease gets smaller with time, but not always monotonically. Therefore care should be taken in selecting the stopping criterion. We have chosen to stop the learning process when the decrease in the cost during the previous 200 sweeps is lower than some predefined threshold.

6.2 Structural Learning and Local Minima

The chosen model has a pre-specified structure which, however, has some flexibility. The number of nodes is not fixed in advance, but their optimal number is estimated using variational Bayesian learning, and unnecessary connections can be pruned away.

A factorial posterior approximation, which is used in this paper, often leads to automatic pruning of some of the connections in the model. When there is not enough data to estimate all the parameters, some directions remain ill-determined. This causes the posterior distribution along those directions to be roughly equal to the prior distribution. In variational Bayesian learning with a factorial posterior approximation, the ill-determined directions tend to get aligned with the axes of the parameter space because then the factorial approximation is most accurate.

The pruning tendency makes it easy to use for instance sparsely connected models, because the learning algorithm automatically selects a small amount of well-determined parameters. But at the early stages of learning, pruning can be harmful, because large parts of the model can get pruned away before a sensible representation has been found. This corresponds to the situation where the learning scheme ends up into a local minimum of the cost function (MacKay, 2001). A posterior approximation which takes into account the posterior dependences has the advantage that it has far less local minima than a factorial posterior approximation. It seems that Bayesian learning algorithms which have linear time complexity cannot avoid local minima in general.

However, suitable choices of the model structure and countermeasures included in the learning scheme can alleviate the problem greatly. We have used the following means for avoiding getting stuck into local minima:

- Learning takes place in several stages, starting from simpler structures which are learned first before proceeding to more complicated hierarchic structures. An example of this technique was presented in Section 5.3.

- New parts of the network are initialised appropriately. One can use for instance principal component analysis (PCA), independent component analysis (ICA), vector quantisation, or kernel PCA (Honkela et al., 2004). The best option depends on the application. Often it is useful to try different methods and select the one providing the smallest value of the cost function for the learned model. There are two ways to handle initialisation: either to fix the sources for a while and learn the weights of the model, or to fix the weights for a while and learn the sources corresponding to the observations. The fixed variables can be released gradually (Valpola et al., 2003c, Section 5.1).

- Automatic pruning is discouraged initially by omitting the term

$$2\mathrm{Var}\{s_2\}\frac{\partial C}{\partial \mathrm{Var}\{s_1 s_2\}}\langle s_1\rangle$$

in the multiplication nodes (Eq. (23)). This effectively means that the mean of s_1 is optimistically adjusted as if there were no uncertainty about s_2. In this way the cost function may increase at first due to overoptimism, but it may pay off later on by escaping early pruning.

- New sources $s_i(t)$ (components of the source vector $\mathbf{s}(t)$ of a layer) are generated, and pruned sources are removed from time to time.

- The activations of the sources are reset a few times. The sources are re-adjusted to their places while keeping the mapping and other parameters fixed. This often helps if some of the sources are stuck into a local minimum.

7. Experimental Results

The Bayes Blocks software (Valpola et al., 2003a) has been applied to several problems.

Valpola et al. (2004) considered several models of variance. The main application was the analysis of MEG measurements from a human brain. In addition to features corresponding to brain activity the data contained several artifacts such as muscle activity induced by the patient biting his teeth. Linear ICA applied to the data was able to separate the original causes to some degree but still many dependencies remained between the sources. Hence an additional layer of so-called variance sources was used to find correlations between the variances of the innovation processes of the ordinary sources. These were able to capture phenomena related to the biting artifact as well as to rhythmic activity.

An astrophysical problem of separating young and old star populations from a set of elliptical galaxy spectra was studied by one of the authors in Nolan et al. (2006). Since the observed quantities were energies and thus positive and since the mixing process was also known to be positive, it was necessary for the subsequent astrophysical analysis to be feasible to include these constraints to the model as well. The standard technique of putting a positive prior on the sources was found to have the unfortunate technical shortcoming of inducing sparsely distributed factors, which was deemed inappropriate in that specific application. To get rid of the induced sparsity but to still keep the positivity constraint, the nonnegativity was forced by rectification nonlinearities (Harva and Kabán, 2007). In addition to finding an astrophysically meaningful factorisation, several other specifications were needed to be met related to handling of missing values, measurements errors and predictive capabilities of the model.

In Raiko (2005), a nonlinear model for relational data was applied to the analysis of the board game Go. The difficult part of the game state evaluation is to determine which groups of stones are likely to get captured. A model similar to the one that will be described in Section 7.2, was built for features of pairs of groups, including the probability of getting captured. When a learned model is applied to new game states, the estimates propagate through a network of such pairs. The structure of the network is thus determined by the game state. The approach can be used for inference in relational databases.

The following three sets of experiments are given as additional examples. The first one is a difficult toy problem that illustrates hierarchy and variance modelling, the second one studies the inference of missing values in speech spectra, and the third one has a dynamical model for image sequences.

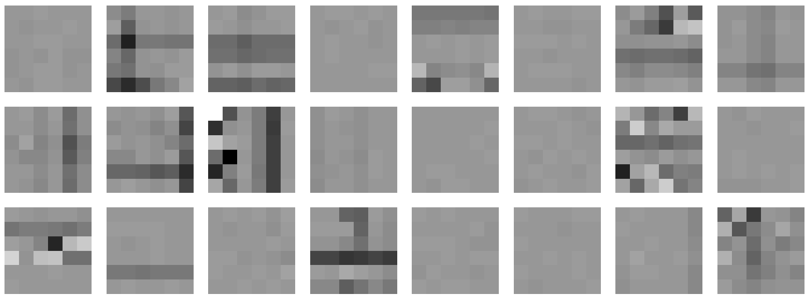


Figure 9: Samples from the 1000 image patches used in the extended bars problem. The bars include both standard and variance bars in horizontal and vertical directions. For instance, the patch at the bottom left corner shows the activation of a standard horizontal bar above the horizontal variance bar in the middle.

7.1 Bars Problem

The first experimental problem studied was testing of the hierarchical nonlinear variance model in Figure 7 in an extension of the well-known bars problem (Dayan and Zemel, 1995). The data set consisted of 1000 image patches each having 6×6 pixels. They contained both horizontal and vertical bars. In addition to the regular bars, the problem was extended to include horizontal and vertical variance bars, characterised and manifested by their higher variance. Samples of the image patches used are shown in Figure 9.

The data were generated by first choosing whether vertical, horizontal, both, or neither orientations were active, each with probability $1/4$. Whenever an orientation is active, there is a probability $1/3$ for a bar in each row or column to be active. For both orientations, there are 6 regular bars, one for each row or column, and 3 variance bars which are 2 rows or columns wide. The intensities (grey level values) of the bars were drawn from a normalised positive exponential distribution having the pdf $p(z) = \exp(-z), z \geq 0$, $p(z) = 0, z < 0$. Regular bars are additive, and variance bars produce additive Gaussian noise having the standard deviation of its intensity. Finally, Gaussian noise with a standard deviation 0.1 was added to each pixel.

The network was built up following the stages shown in Figure 7. It was initialised with a single layer with 36 nodes corresponding to the 36 dimensional data vector. The second layer of 30 nodes was created at the sweep 20, and the third layer of 5 nodes at the sweep 100. After creating a layer only its sources were updated for 10 sweeps, and pruning was discouraged for 50 sweeps. New nodes were added twice, 3 to the second layer and 2 to the third layer, at sweeps 300 and 400. After that, only the sources were updated for 5 sweeps, and pruning was again discouraged for 50 sweeps. The source activations were reset at the sweeps 500, 600 and 700, and only the sources were updated for the next 40 sweeps. Dead nodes were removed every 20 sweeps. The multistage training procedure was designed to avoid suboptimal local solutions, as discussed in Section 6.2. Note that a large part of the presented procedure could be automated for application to any problem.

Figure 10 demonstrates that the algorithm finds a generative model that is quite similar to the generation process. The two sources on the third layer correspond to the horizontal and vertical orientations and the 18 sources on the second layer correspond to the bars. Each element of the weight matrices is depicted as a pixel with the appropriate grey level value in Fig. 10. The pixels of $\mathbf{A}_2$ and $\mathbf{B}_2$ are ordered similarly as the patches of $\mathbf{A}_1$ and $\mathbf{B}_1$, that is, vertical bars on the left and horizontal bars on the right. Regular bars, present in the mixing matrix $\mathbf{A}_1$, are reconstructed accurately, but the variance bars in the mixing matrix $\mathbf{B}_1$ exhibit some noise. The grouping of horizontal and vertical orientations is clearly visible in the mixing matrix $\mathbf{A}_2$. For instance, the first patch in $\mathbf{A}_2$ has non-zero (black) weights to all the vertical features, that is, left side pixels in $\mathbf{A}_2$ activate sources corresponding to the left side patches in $\mathbf{A}_1$ and $\mathbf{B}_1$.

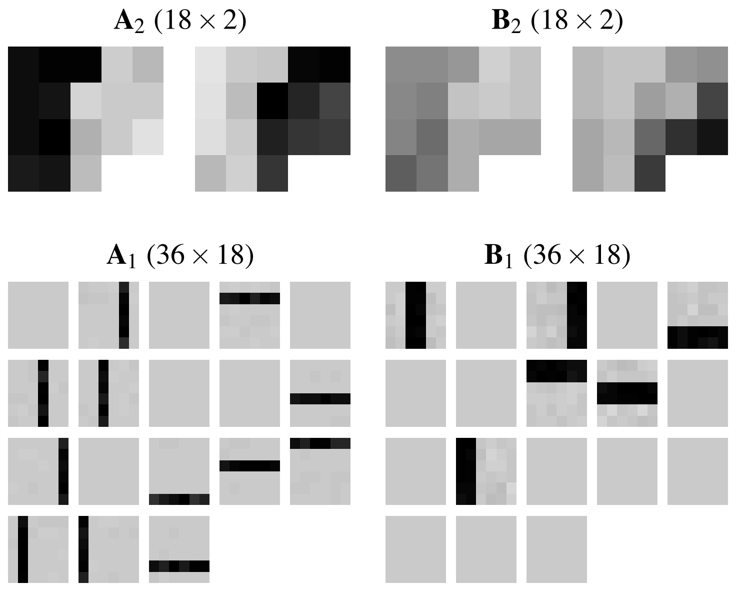
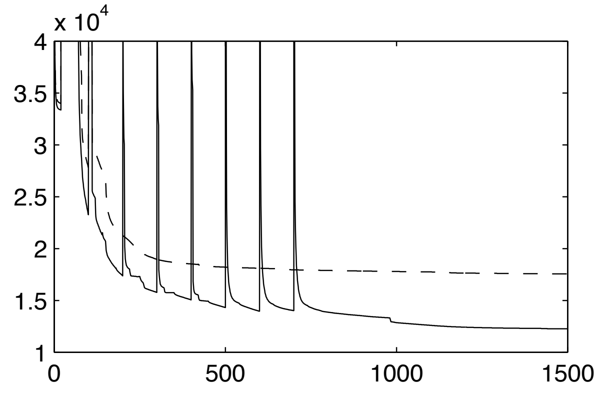


Figure 10: Results of the extended bars problem: Posterior means of the weight matrices after learning. The sources of the second layer have been ordered for visualisation purposes according to the weight (mixing) matrices $\mathbf{A}_2$ and $\mathbf{B}_2$. The elements of the matrices have been depicted as pixels having corresponding grey level values. The 18 pixels in the weight matrices $\mathbf{A}_2$ and $\mathbf{B}_2$ correspond to the 18 patches in the weight matrices $\mathbf{A}_1$ and $\mathbf{B}_1$.

A comparison experiment with a simplified learning procedure was run to demonstrate the importance of local optima. The creation and pruning of layers were done as before, but other methods for avoiding local minima (addition of nodes, discouraging pruning and resetting of sources) were disabled. The resulting weights can be seen in Figure 11. This time the learning ends up in a suboptimal local optimum of the cost function. One of the bars was not found (second horizontal bar from the bottom), some were mixed up in a same source (most variance bars share a source with a regular bar), fourth vertical bar from the left appears twice, and one of the sources just suppresses variance everywhere. The resulting cost function (4) is worse by 5292 compared to the main experiment. The ratio of the model evidences is thus roughly $\exp(5292)$.

Figure 12 illustrates the formation of the posterior distribution of a typical single variable. It is the first component of the variance source $\mathbf{u}_1(1)$ in the comparison experiment. The prior means here the distribution given its parents (especially $\mathbf{s}_2(1)$ and $\mathbf{B}_1$) and the likelihood means the po-



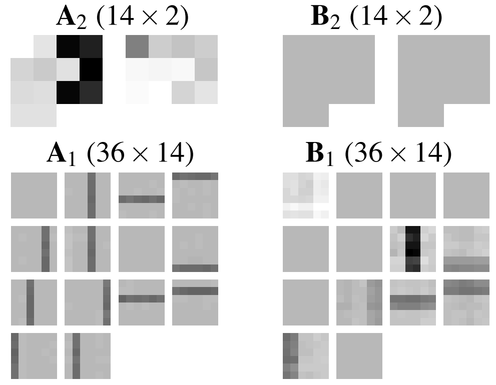


Figure 11: *Left:* Cost function plotted against the number of learning sweeps. Solid curve is the main experiment and the dashed curve is the comparison experiment. The peaks appear when nodes are added. *Right:* The resulting weights in the comparison experiment are plotted like in Figure 10.

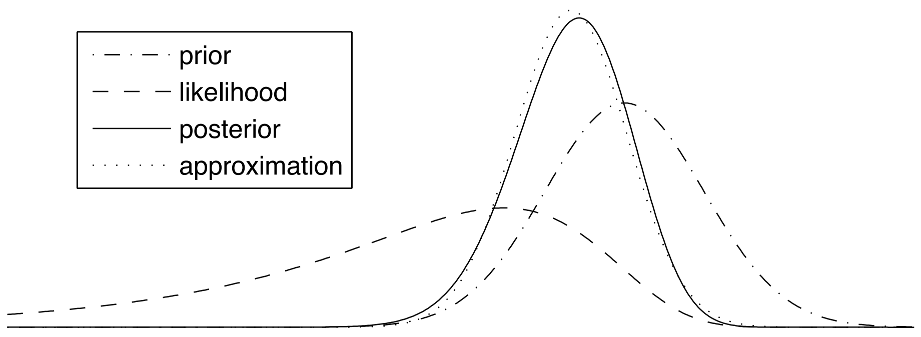


Figure 12: A typical example illustrating the posterior approximation of a variance source.

tential given its children (the first component of $\mathbf{x}(1)$). Assuming the posteriors of other variables accurate, we can plot the true posterior of this variable and compare it to the Gaussian posterior approximation. Their difference is only 0.007 measured by Kullback-Leibler divergence.

7.2 Missing Values in Speech Spectra

In hierarchical nonlinear factor analysis (HNFA) (Valpola et al., 2003c), there are a number of layers of Gaussian variables, the bottom-most layer corresponding to the data. There is a nonlinearity and a linear mixture mapping from each layer to all the layers below it.

HNFA resembles the model structure in Section 5.3. The model structure is depicted in the left subfigure of Fig. 13. Model equations are

$$\begin{aligned}\mathbf{h}(t) &= \mathbf{A}\mathbf{s}(t) + \mathbf{a} + \mathbf{n}_h(t),\\ \mathbf{x}(t) &= \mathbf{B}\phi[\mathbf{h}(t)] + \mathbf{C}\mathbf{s}(t) + \mathbf{b} + \mathbf{n}_x(t),\end{aligned}$$

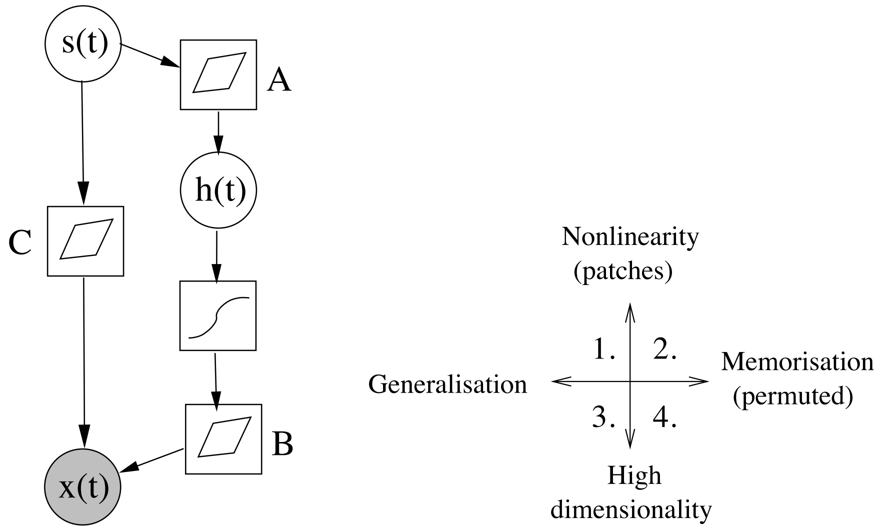


Figure 13: *Left:* The model structure for hierarchical nonlinear factor analysis (HNFA). *Right:* Four different experimental settings with the speech data used for measuring different properties of the algorithms.

where $\mathbf{n}_h(t)$ and $\mathbf{n}_x(t)$ are Gaussian noise terms and the nonlinearity $\phi(\xi) = \exp(-\xi^2)$ again operates on each element of its argument vector separately. Note that we have included a short-cut mapping $\mathbf{C}$ from sources to observations. This means that hidden nodes only need to model the deviations from linearity.

HNFA is compared against three other methods. Factor analysis (FA) is a linear method described in Section 5.2. It is a special case of HNFA where the dimensionality of $\mathbf{h}(t)$ is zero. Nonlinear factor analysis (NFA) (Lappalainen and Honkela, 2000; Honkela and Valpola, 2005) differs from HNFA in that it does not use mediating variables $\mathbf{h}(t)$:

$$\mathbf{x}(t) = \mathbf{B}\tanh[\mathbf{A}\mathbf{s}(t) + \mathbf{a}] + \mathbf{b} + \mathbf{n}_x(t).$$

Note that NFA has multiple computational paths between $\mathbf{s}(t)$ and $\mathbf{x}(t)$, which leads to a higher computational complexity compared to HNFA.

The self-organising map SOM (Kohonen, 2001) differs most from the other methods. A rectangular map has a number of map units with associated model vectors that are points in the data space. Each data point is matched to the closest map unit. The model vectors of the best-matching unit and its neighbours in the map are moved slightly towards the data point. See Kohonen (2001); Haykin (1998) for details.

The data set consisted of speech spectrograms from several Finnish subjects. Short term spectra were acquired by using Fourier transformation and their dimensionality was reduced to 30 by triangular windowing with window centres uniformly spaced on the Mel scale. It is clear that a dynamic source model would give better reconstructions, but in this case the temporal information was left out to ease the comparison of the models. Half of the about 5000 samples were used as test

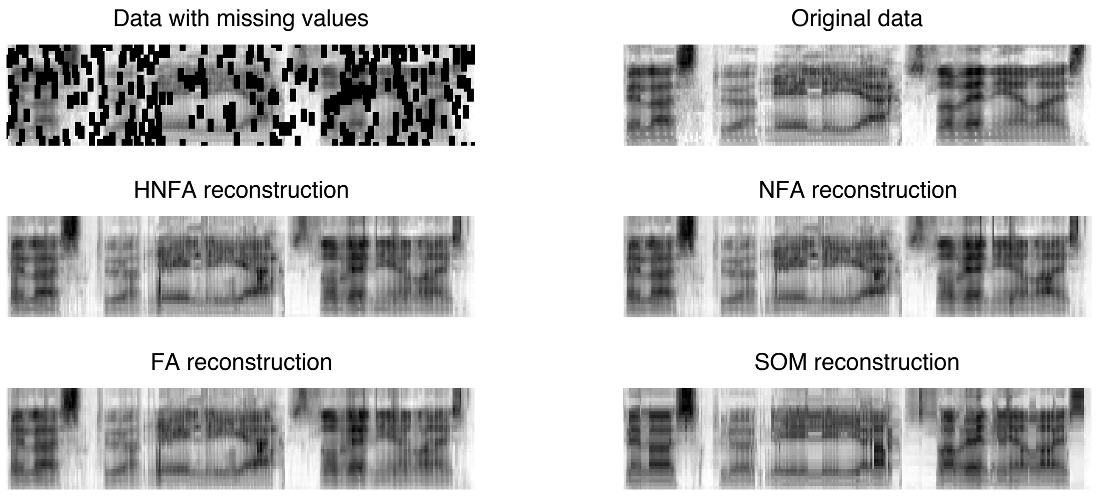


Figure 14: Some speech data with and without missing values (Setting 1) and reconstructions by HNFA and by comparison models.

data with some missing values. Missing values were set in four different ways to measure different properties of the algorithms (right subfigure of Fig. 13):

1. 38 percent of the values are set to miss randomly in 4×4 patches. (Figure 14)
2. The data samples are randomly permuted before setting missing values in 4×4 patches as in Setting 1.
3. 10 percent of the values are set to miss randomly independent of any neighbours. This is an easier setting, since simple smoothing using nearby values would give fine reconstructions.
4. The data samples are permuted and 10 percent of the values are set to miss independently of any neighbours.

With permutation, data samples from a single spoken word end up in both training and testing data. This setting emphasises memorisation, while testing with new speech emphasises generalisation. When values are missing one by one, they could be reconstructed quite well even by linear interpolation of their neighbours. Thus, this setting emphasises the models ability to work accurately in high dimensions rather than its capability of modelling nonlinear effects.

We tried to optimise each method and in the following, we describe how we got the best results. The self-organising map was run using the SOM Toolbox (Vesanto et al., 1999) with long learning time, 2500 map units and random initialisations. In other methods, the optimisation was based on minimising the cost function or its approximation. NFA was learned for 5000 sweeps through data using a Matlab implementation. Varying number of sources were tried out and the best ones were used as the result. The optimal number of sources was around 12 to 15 and the size used for the hidden layer was 30. A large enough number should do, since the algorithm can effectively prune out parts that are not needed.

In factor analysis (FA), the number of sources was 28. In hierarchical nonlinear factor analysis (HNFA), the number of sources at the top layer was varied and the best runs according to the cost function were selected. In those runs, the size of the top layer varied from 6 to 12 and the size of the middle layer, which is determined during learning, turned out to vary from 12 to 30. HNFA was run for 5000 sweeps through data. Each experiment with NFA or HNFA took about 8 hours of processor time, while FA and SOM were faster.

Several runs were conducted with different random initialisations but with the same data and the same missing value pattern for each setting and for each method. The number of runs in each cell is about 30 for HNFA, 4 for NFA and 20 for the SOM. FA always converges to the same solution. The mean and the standard deviation of the mean square reconstruction error of missing values are:

	FA	HNFA	NFA	SOM
Setting 1	1.87	1.80 ± 0.03	1.74 ± 0.02	1.69 ± 0.02
Setting 2	1.85	1.78 ± 0.03	1.71 ± 0.01	1.55 ± 0.01
Setting 3	0.57	$0.55 \pm .005$	$0.56 \pm .002$	0.86 ± 0.01
Setting 4	0.58	$0.55 \pm .008$	$0.58 \pm .004$	0.87 ± 0.01

The order of results of the Setting 1 follow our expectations on the nonlinearity of the models. The SOM with highest nonlinearity gives the best reconstructions, while NFA, HNFA and finally FA follow in that order. Example reconstructions are shown in Figure 14. The results of HNFA vary the most - there is potential to develop better learning schemes to find better solutions more often. The sources $\mathbf{h}(t)$ of the hidden layer did not only emulate computation nodes, but they were also active themselves. Avoiding this situation during learning could help to find more nonlinear and thus perhaps better solutions.

In the Setting 2, due to the permutation, the test set contains vectors very similar to some in the training set. Therefore, generalisation is not as important as in the Setting 1. The SOM is able to memorise details corresponding to individual samples better due to its high number of parameters. Compared to the Setting 1, SOM benefits a lot and makes clearly the best reconstructions, while the others benefit only marginally.

The Settings 3 and 4, which require accurate expressive power in high dimensionality, turned out not to differ from each other much. The basic SOM has only two intrinsic dimensions[3] and therefore it was clearly poorer in accuracy. Nonlinear effects were not important in these settings, since HNFA and NFA were only marginally better than FA. HNFA was better than NFA perhaps because it had more latent variables when counting both $\mathbf{s}(t)$ and $\mathbf{h}(t)$.

To conclude, HNFA lies between FA and NFA in performance. HNFA is applicable to high dimensional problems and the middle layer can model part of the nonlinearity without increasing the computational complexity dramatically. FA is better than SOM when expressivity in high dimensions is important, but SOM is better when nonlinear effects are more important. The extensions of FA, NFA and HNFA, expectedly performed better than FA in each setting. It may be possible to enhance the performance of NFA and HNFA by new learning schemes whereas especially FA is already at its limits. On the other hand, FA is best if low computational complexity is the determining factor.

7.3 Variance Model of Image Sequences

In this section an experiment with a dynamical model for variances applied to image sequence analysis is reported. The motivation behind modelling variances is that in many natural signals,

3. Higher dimensional SOMs become quickly intractable due to exponential number of parameters.

there exists higher order dependencies which are well characterised by correlated variances of the signals (Parra et al., 2001). Hence we postulate that we should be able to better catch the dynamics of a video sequence by modelling the variances of the features instead of the features themselves. This indeed is the case as will be shown.

The model considered can be summarised by the following set of equations:

$$\begin{aligned}\mathbf{x}(t) &\sim \mathcal{N}(\mathbf{A}\mathbf{s}(t), \operatorname{diag}(\exp[-\mathbf{v}_x])),\\ \mathbf{s}(t) &\sim \mathcal{N}(\mathbf{s}(t-1), \operatorname{diag}(\exp[-\mathbf{u}(t)])),\\ \mathbf{u}(t) &\sim \mathcal{N}(\mathbf{B}\mathbf{u}(t-1), \operatorname{diag}(\exp[-\mathbf{v}_u])).\end{aligned}$$

We will use the acronym DynVar in referring to this model. The linear mapping $\mathbf{A}$ from sources $\mathbf{s}(t)$ to observations $\mathbf{x}(t)$ is constrained to be sparse by assigning each source a circular region on the image patch outside of which no connections are allowed. These regions are still highly overlapping. The variances $\mathbf{u}(t)$ of the innovation process of the sources have a linear dynamical model. It should be noted that modelling the variances of the sources in this manner is impossible if one is restricted to use conjugate priors.

The sparsity of $\mathbf{A}$ is crucial as the computational complexity of the learning algorithm depends on the number of connections from $\mathbf{s}(t)$ to $\mathbf{x}(t)$. The same goal could have been reached with a different kind of approach as well. Instead of constraining the mapping to be sparse from the very beginning of learning it could have been allowed to be full for a number of iterations and only after that pruned based on the cost function as explained in Section 6.2. But as the basis for image sequences tends to get sparse anyway, it is a waste of computational resources to wait while most of the weights in the linear mapping tend to zero.

For comparison purposes, we postulate another model where the dynamical relations are sought directly between the sources leading to the following model equations:

$$\begin{aligned}\mathbf{x}(t) &\sim \mathcal{N}(\mathbf{A}\mathbf{s}(t), \operatorname{diag}(\exp[-\mathbf{v}_x])),\\ \mathbf{s}(t) &\sim \mathcal{N}(\mathbf{B}\mathbf{s}(t-1), \operatorname{diag}(\exp[-\mathbf{v}_s])).\end{aligned}$$

We shall refer to this model as DynSrc.

The data $\mathbf{x}(t)$ was a video image sequence (van Hateren and Ruderman, 1998) of dimensions $16 \times 16 \times 4000$. That is, the data consisted of 4000 subsequent digital images of the size 16×16. A part of the data set is shown in Figure 15.

Both models were learned by iterating the learning algorithm 2000 times at which stage a sufficient convergence was attained. The first hint of the superiority of the DynVar model was provided by the difference of the cost between the models which was 28 bits/frame (for the coding interpretation, see Honkela and Valpola, 2004). To further evaluate the performance of the models, we considered a simple prediction task where the next frame was predicted based on the previous ones. The predictive distributions, $p(\mathbf{x}(t+1)|\mathbf{x}(1),...,\mathbf{x}(t))$, for the models can be approximately computed based on the posterior approximation. The means of the predictive distributions are very similar for both of the models. Figure 16 shows the means of the DynVar model for the same sequence as in Figure 15. The means themselves are not very interesting, since they mainly reflect the situation in the previous frame. However, the DynVar model provides also a rich model for the variances. The standard deviations of its predictive distribution are shown in Figure 17. White stands for a large variance and black for a small one. Clearly, the model is able to increase the predicted variance in the area of high motion activity and hence provide better predictions. We can offer quantitative support for this claim by computing the predictive perplexities for the models. Predictive perplexity is widely used in language modelling and it is defined as

$$\text{perplexity}(t) = \exp\left\{ -\frac{1}{256} \sum_{i=1}^{256} \log p(x_i(t+1)|\mathbf{x}(1), ..., \mathbf{x}(t)) \right\}.$$

The predictive perplexities for the same sequence as in Figure 15 are shown in Figure 18. Naturally the predictions get worse when there is movement in the video. However, DynVar model is able to handle it much better than the compared DynSrc model. The same difference can also be directly read by comparing the cost functions (2).

The possible applications for a model of image sequences include video compression, motion detection, early stages of computer vision, and making hypotheses on biological vision.

8. Discussion

One of the distinctive factors between different Bayesian approaches is the type of posterior approximation. We have concentrated on large unsupervised learning tasks, where point estimates are too prone to overfitting and sampling used in MCMC methods and particle filters, is often too slow. The problems tackled with particle filters in Doucet et al. (2001) vary from 1 to 10 in dimensionality, whereas the latent space in Section 7.3 is 128 dimensional. The variational Bayesian learning seems to provide a good compromise between point estimates and sampling methods.

Often the posterior distribution consists of clusters or solution modes. It depends on the posterior approximation again, whether only one of the clusters, or all of them are modelled. In our case, the expectation in $\mathcal{J}_{\mathrm{KL}}(q \parallel p)$ is taken over the approximate distribution q, which in practice leads to modelling a single mode. In expectation propagation (Minka, 2001), the Kullback-Leibler divergence is formed differently, leading to modelling of all the modes. Also, sampling is supposed to take place in all the modes. For the purpose of finding a single good representative of the posterior probability mass, the first approach should be better. In fact, the expectation over the true posterior, also known as the Bayes estimate, is often degenerate due to symmetry. For instance in factor analysis type models, the posterior is symmetric to the permutation of factors. The number of permutations also gives a hint of the infeasibility of accurately modelling all the modes in a high-dimensional problem. Perhaps it would be best to find one mode for the parameters, but all modes for the time-dependent variables when feasible.

The variational Bayesian methods vary further depending on the posterior approximation. In this paper, all variables are assumed to be independent a posteriori. We have chosen to model individual distributions as Gaussians. Often different conjugate distributions are used instead, for instance, the variance of a Gaussian variable is modelled with a Gamma distribution. Conjugate distributions are accurate and in some sense practical, but by restricting to Gaussians, the nodes can be connected more freely allowing for example hierarchical modelling of variances. It should be noted that the effect of assuming independencies is far more significant compared to the effect of approximations in modelling individual distributions.

The scope of this paper has been restricted to models which can be learned using purely local computations. This is possible, if the parents of each node are independent a posteriori. This can be accomplished by using a factorial posterior approximation and by not allowing multiple computational paths between variables. Purely local computations result in a computational complexity that is linear w.r.t. the number of connections in the model. In small models, one could afford to

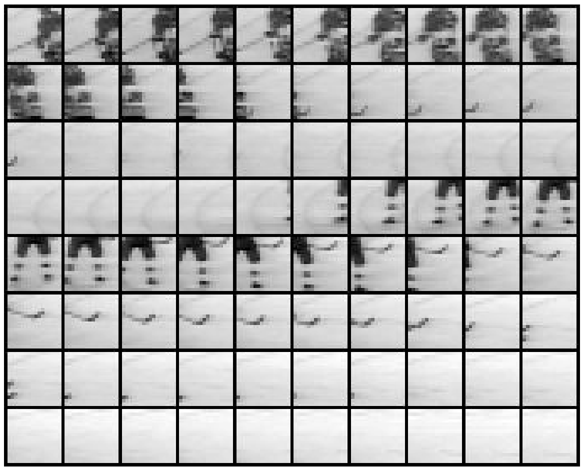


Figure 15: A sequence of 80 frames from the data used in the experiment.

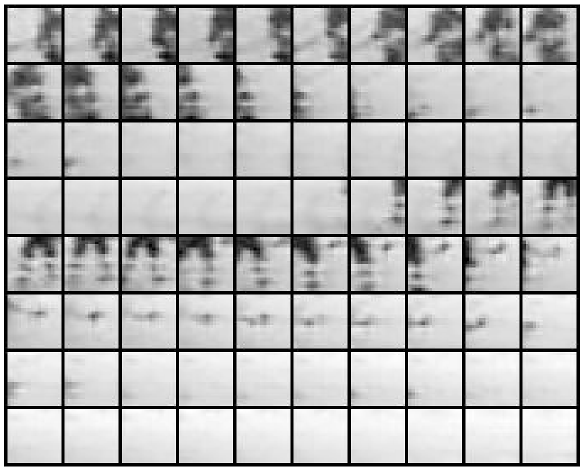


Figure 16: The means of the predictive distribution for the DynVar model.

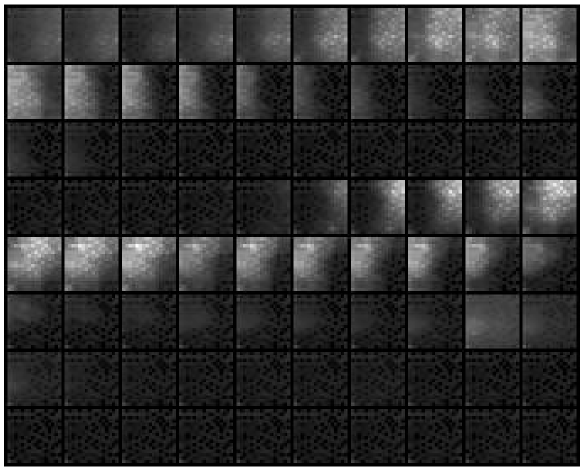


Figure 17: The standard deviations of the predictive distribution for the DynVar model.

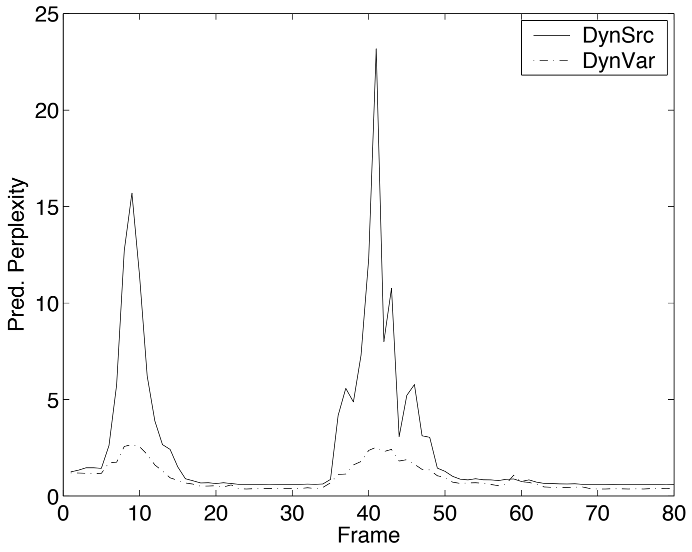


Figure 18: Predictive perplexities. Perplexity means that the model is as perplexed as if it were choosing between perplexity(t) equally likely options. Low perplexity means high compression.

take all the dependencies into account. In larger models, it might be desirable to model posterior dependencies within disjoint groups of variables, but to assume the groups statistically independent, as is done by Winn and Bishop (2005).

According to our experience, almost maximally factorial posterior pdf approximation $q(\theta)$ suffices in many cases. It seems that a good model structure is usually more important than a good approximation of the posterior pdf of the model. Therefore the available computation time is often better invested in a larger model using a simple posterior approximation. In any case, density estimates of continuous valued latent variables offer an important advantage over point estimates, because they are robust against overfitting and provide a cost function suitable for learning model structures. With variational Bayesian learning employing a factorial posterior pdf approximation $q(\theta)$ the density estimates are almost as efficient as point estimates. Moreover, latent variable models often exhibit rotational and other invariances of which variational Bayesian learning can take advantage by choosing a solution where the factorial approximation is most accurate.

The basic algorithm for learning and inference is based on updating a variable at a time while keeping other variables fixed. It has the benefits of being completely local and guaranteed to converge. A drawback is that the flow of information through time can be slow while performing inference in a dynamical model. There are alternative inference algorithms, where updates are carried out in forward and backward sweeps. These include particle smoothing (Doucet et al., 2001), extended Kalman smoothing (Anderson and Moore, 1979), and expectation propagation (Minka, 2001). Raiko et al. (2006) propose an algorithm that combines guaranteed convergence with faster flow of information in variational Bayesian inference that could be applied here as well. When the model needs to be learned at the same time, one needs to iterate a lot anyway, so the variational Bayesian algorithm that makes small but consistent improvements at every sweep might be preferable.

It is an important design choice that each node is updated while keeping the other nodes fixed. If new node types are added later on, there is no need to change the global learning algorithm, but it suffices to design an update rule for the new node type. Also, there is an option to update some nodes more often than others. When different parameters are coupled and cyclic updating is slow, it can be sped up by line search as described by Honkela et al. (2003). Note that all the updates are done in order to minimise a global cost function, that is, a cost function over all the variables. Expectation propagation (Minka, 2001) updates one approximation at a time, too. An important difference is that there updates are done using a local cost function, that is, the local approximation is fitted to the local true posterior assuming that the rest of the approximation is accurate. This is the reason why expectation propagation may diverge.

Large nonlinear problems often have numerous suboptimal local solutions that should be avoided. We have used many tricks to avoid them, as discussed in Section 6.2. It depends on the application which tricks work best. It is an important aspect of future work to make the procedure as simple as possible for the user.

9. Conclusions

In this paper, we have introduced standardised nodes (blocks) for constructing generative latent variable models. These nodes include a Gaussian node, addition, multiplication, a nonlinearity following directly a Gaussian node, and a delay node. The nodes have been designed so that they fit together, allowing construction of many types of latent variable models, including both known and novel structures. Constructing new prototype models is rapid since the user does not need to take care of the learning formulas. The nodes have been implemented in an open source software package called the Bayes Blocks (Valpola et al., 2003a).

The models built from these blocks are taught using variational Bayesian (ensemble) learning. This learning method essentially uses as its cost function the Kullback-Leibler information between the true posterior density and its approximation. The cost function is used for updating the unknown variables in the model, but it also allows optimisation of the number of nodes in the chosen model type. By using a factorial posterior density approximation, all the required computations can be carried out locally by propagating means, variances, and expected exponentials instead of full distributions. In this way, one can achieve a linear computational complexity with respect to the number of connections in the chosen model. However, initialisation to avoid premature pruning of nodes and local minima require special attention in each application for achieving good results.

In this paper, we have tested the introduced method experimentally in three separate unsupervised learning problems with different types of models. The results demonstrate the good performance and usefulness of the method. First, hierarchical nonlinear factor analysis (HNFA) with variance modelling was applied to an extension of the bars problem. The presented algorithm could find a model that is essentially the same as the complicated way in which the data were generated. Secondly, HNFA was used to reconstruct missing values in speech spectra. The results were consistently better than with linear factor analysis, and were generally best in cases requiring accurate representation in high dimensionality. The third experiment was carried out using real-world video image data. We compared the linear dynamical model for the means and for the variances of the sources. The results demonstrate that finding strong dependencies between different sources was considerably easier when the variances were modelled, too.

Acknowledgments

We thank Antti Honkela, Tomas Östman, and Alexander Ilin for work on Bayes blocks software library and for useful comments, and Hans van Hateren for supplying the video data (van Hateren and Ruderman, 1998) used in the experiments. This research has been funded by the European Commission project BLISS and the Finnish Centre of Excellence Programme (2000-2005) under the project New Information Processing Principles.

Appendix A. Updating $q(s)$ for the Gaussian Node

Here we show how to minimise the function

$$\mathcal{C}(m,v) = Mm + V[(m-m_0)^2 + v] + E\exp(m+v/2) - \frac{1}{2}\ln v,$$

where M,V,E, and m_0 are scalar constants. A unique solution exists when $V > 0$ and $E \geq 0$. This problem occurs when a Gaussian posterior with mean m and variance v is fitted to a probability distribution whose logarithm has both a quadratic and exponential part resulting from Gaussian prior and log-Gamma likelihoods, respectively, and Kullback-Leibler divergence is used as the measure of the misfit.

In the special case $E = 0$, the minimum of $\mathcal{C}(m,v)$ can be found analytically and it is $m = m_0 - \frac{M}{2V}$, $v = \frac{1}{2V}$. In other cases where $E > 0$, minimisation is performed iteratively. At each iteration, one Newton iteration for the mean m and one fixed-point iteration for the variance v are carried out as explained in more detail in the following.

A.1 Newton Iteration for the Mean m

The Newton iteration for m is obtained by

$$\begin{aligned} m_{i+1} &= m_i - \frac{\partial C(m_i, v_i)/\partial m_i}{\partial^2 C(m_i, v_i)/\partial m_i^2} \\ &= m_i - \frac{M + 2V(m_i - m_0) + E\exp(m + v/2)}{2V + E\exp(m + v/2)}. \end{aligned} \tag{38}$$

The Newton iteration converges in one step if the second derivative remains constant. The step is too short if the second derivative decreases and too long if the second derivative increases. For stability, it is better to take too short than too long steps.

In this case, the second derivative always decreases if the mean m decreases and vice versa. For stability it is therefore useful to restrict the growth of m because it is consistently over-estimated.

A.2 Fixed-Point Iteration for the Variance v

A simple fixed-point iteration rule is obtained for the variance v by solving the zero of the derivative:

$$\begin{aligned} 0 &= \frac{\partial C(m, v)}{\partial v} = V + \frac{E}{2}\exp(m + v/2) - \frac{1}{2v} \Leftrightarrow \\ v &= \frac{1}{2V + E\exp(m + v/2)} \overset{def}{=} g(v), \\ &\qquad v_{i+1} = g(v_i). \end{aligned} \tag{39}$$

In general, fixed-point iterations are stable around the solution v_{opt} if $|g'(v_{\text{opt}})| < 1$ and converge best when the derivative $g'(v_{\text{opt}})$ is near zero. In our case $g'(v_i)$ is always negative and can be less than -1. In this case the solution can be an unstable fixed-point. This can be avoided by taking a weighted average of (39) and a trivial iteration $v_{i+1} = v_i$:

$$v_{i+1} = \frac{\xi(v_i)g(v_i) + v_i}{\xi(v_i) + 1} \overset{def}{=} f(v_i). \tag{40}$$

The weight ξ should be such that the derivative of f is close to zero at the optimal solution v_{opt} which is achieved exactly when $\xi(v_{\text{opt}}) = -g'(v_{\text{opt}})$.

It holds

$$\begin{aligned} g'(v) &= -\frac{(E/2)\exp(m + v/2)}{[2V + E\exp(m + v/2)]^2} = g^2(v)\left[V - \frac{1}{2g(v)}\right] = g(v)\left[Vg(v) - \frac{1}{2}\right] \Rightarrow \\ g'(v_{\text{opt}}) &= v_{\text{opt}}\left[Vv_{\text{opt}} - \frac{1}{2}\right] \Rightarrow \xi(v_{\text{opt}}) = v_{\text{opt}}\left[\frac{1}{2} - Vv_{\text{opt}}\right]. \end{aligned}$$

The last steps follow from the fact that $v_{\text{opt}} = g(v_{\text{opt}})$ and from the requirement that $f'(v_{\text{opt}}) = 0$. We can assume that v is close to v_{opt} and use

$$\xi(v) = v\left[\frac{1}{2} - Vv_{\text{opt}}\right]. \tag{41}$$

Note that the iteration (39) can only yield estimates with $0 < v_{i+1} < 1/2V$ which means that $\xi(v_{i+1}) > 0$. Therefore the use of ξ always shortens the step taken in (40). If the initial estimate $v_0 > 1/2V$, we can set it to $v_0 = 1/2V$.

There is potential for a speedup by doing a joint Newton's iteration on both m and v at the same time. Another option would be to allow m to be updated more often than v.

A.3 Summary of the Updating Method for $q(s)$

1. Set $v_0 \leftarrow \min(v_0, 1/2V)$.
2. Iterate:
   (a) Solve the new estimate of the mean m from Eq. (38) under the restriction that the maximum step is 4;
   (b) Solve the new estimate of the variance v from Eqs. (41) and (40) under the restriction that the maximum step is 4.
3. Stop the iteration after the corrections become small enough, being under some suitable predefined threshold value.

Appendix B. Addition and Multiplication Nodes

Equations (15)–(19) for the addition and multiplication nodes are proven in the following section. Only the Equation (15) applies in general, the others assume that the incoming signals are independent a posteriori. That is, $q(s_1, s_2, \ldots, s_n) = q(s_1)q(s_2)\ldots q(s_n)$. Also the proof of the form of the cost function mostly concerns propagation through addition and multiplication nodes, so it is presented here. Finally, the formulas of propagating the gradients of the cost function w.r.t. the expectations are derived.

B.1 Expectations

Equation (15) follows directly from the linearity of the expectation operation, or can be proven analogously to the proof of Equation (18):

$$\begin{aligned}\left\langle \prod_{i=1}^{n} s_i \right\rangle &= \int \left(\prod_{i=1}^{n} s_i \right) q(s_1, s_2, \ldots, s_n) d\mathbf{s} \\ &= \int \prod_{i=1}^{n} s_i q(s_i) d\mathbf{s} = \prod_{i=1}^{n} \int s_i q(s_i) ds_i = \prod_{i=1}^{n} \langle s_i \rangle .\end{aligned}$$

Equation (16) states that the variance of a sum of independent variables is the sum of their variances. This fact can be found in basic probability theory books. It can be proven with simple manipulation by using Equations (15) and (18).

Equation (17) can be proven by applying (18) to $\exp s_i$:

$$\left\langle \exp\left(\sum_{i=1}^{n} s_i \right) \right\rangle = \left\langle \prod_{i=1}^{n} \exp s_i \right\rangle = \prod_{i=1}^{n} \langle \exp s_i \rangle .$$

Equation (19) can be proven by applying Equation (18) to both s_i and s_i^2:

$$\begin{aligned}\operatorname{Var}\left\{\prod_{i=1}^n s_i\right\} &= \left\langle\left(\prod_{i=1}^n s_i\right)^2\right\rangle - \left\langle\prod_{j=1}^n s_j\right\rangle^2 = \left\langle\prod_{i=1}^n s_i^2\right\rangle - \left(\prod_{j=1}^n \langle s_j\rangle\right)^2 \\ &= \prod_{i=1}^n \langle s_i^2\rangle - \prod_{j=1}^n \langle s_j\rangle^2 = \prod_{i=1}^n \left[\langle s_i\rangle^2 + \operatorname{Var}\{s_i\}\right] - \prod_{j=1}^n \langle s_j\rangle^2 .\end{aligned}$$

B.2 Form of the Cost Function

The form of the part of the cost function that an output of a node affects is shown to be of the form

$$\mathcal{C}_p = M\langle\cdot\rangle + V[(\langle\cdot\rangle - \langle\cdot\rangle_{\text{current}})^2 + \operatorname{Var}\{\cdot\}] + E\langle\exp\cdot\rangle + C. \tag{42}$$

where $\langle\cdot\rangle$ denotes the expectation of the quantity in question. If the output is connected directly to another variable, this can be seen from Eq. (8) by substituting

$$\begin{aligned} M &= \langle\exp v\rangle\left(\langle s\rangle_{\text{current}} - \langle m\rangle\right), \\ V &= \frac{1}{2}\langle\exp v\rangle, \\ E &= 0, \\ C &= \frac{1}{2}\left[\langle\exp v\rangle\left(\operatorname{Var}\{m\} + \langle m\rangle^2 - \langle s\rangle^2_{\text{current}}\right) - \langle v\rangle + \ln 2\pi\right]. \end{aligned}$$

If the output is connected to multiple variables, the sum of the affected costs is of the same form. Now one has to prove that this form remains the same when the signals are fed through the addition and multiplication nodes.[4]

If the cost function is of the predefined form (42) for the sum $s_1 + s_2$, it has the same form for s_1, when s_2 is regarded as a constant. This can be shown using Eqs. (15), (16), and (17):

$$\begin{aligned}\mathcal{C}_p &= M\langle s_1 + s_2\rangle + V\left[(\langle s_1 + s_2\rangle - \langle s_1 + s_2\rangle_{\text{current}})^2 + \operatorname{Var}\{s_1 + s_2\}\right] \\ &\quad + E\langle\exp(s_1 + s_2)\rangle + C \\ &= M\langle s_1\rangle + V\left[(\langle s_1\rangle - \langle s_1\rangle_{\text{current}})^2 + \operatorname{Var}\{s_1\}\right] \\ &\quad + (E\langle\exp s_2\rangle)\langle\exp s_1\rangle + (C + M\langle s_2\rangle + V\operatorname{Var}\{s_2\}).\end{aligned} \tag{43}$$

It can also be seen from (43) that when $E = 0$ for the sum $s_1 + s_2$, it is zero for the addend s_1, that is $E' = E\langle\exp s_2\rangle = 0$. This means that the outputs of product and nonlinear nodes can be fed through addition nodes.

If the cost function is of the predefined form (42) with $E = 0$ for the product $s_1 s_2$, it is similar for the variable s_1, when the variable s_2 is regarded as a constant. This can be shown using Eqs. (18)

4. Note that delay node only rewires connections so it does not affect the formulas.

and (19):

$$\begin{aligned}
\mathcal{C}_p &= M\langle s_1s_2\rangle + V\left[(\langle s_1s_2\rangle - \langle s_1s_2\rangle_{\text{current}})^2 + \text{Var}\{s_1s_2\}\right] + C \\
&= (M\langle s_2\rangle + 2V\text{Var}\{s_2\}\langle s_1\rangle_{\text{current}})\langle s_1\rangle \\
&\quad + \left[V\left(\langle s_2\rangle^2 + \text{Var}\{s_2\}\right)\right]\left[(\langle s_1\rangle - \langle s_1\rangle_{\text{current}})^2 + \text{Var}\{s_1\}\right] \\
&\quad + \left(C - V\text{Var}\{s_2\}\langle s_1\rangle^2_{\text{current}}\right).
\end{aligned}$$

Appendix C. Updating $q(s)$ for the Gaussian Node Followed by a Nonlinearity

A Gaussian variable has its own terms in the cost function and it affects the cost function of its children. In case there is a nonlinearity attached to it, only the latter is changed. The cost function of the children can be written in the form

$$\mathcal{C}_{\text{ch}(s),p} = M\langle f(s)\rangle + V[(\langle f(s)\rangle - \langle f(s)\rangle_{\text{current}})^2 + \text{Var}\{f(s)\}]$$

where $\langle f(s)\rangle_{\text{current}}$ stands for the expectation using the current posterior estimate $q(s)$, and M and V are constants.

The posterior $q(s) = \mathcal{N}(s;\bar{s},\tilde{s})$ is updated to minimise the cost function. For $\tilde{s}$ we get a fixed point iteration for the update candidate:

$$\tilde{s}_{new} = \left[\langle\exp v\rangle + \frac{4V\left(1-2\bar{s}^2+2\tilde{s}\right)\left(\langle f(s)\rangle - \frac{M}{2V}\right)\langle f(s)\rangle}{(2\tilde{s}+1)^2} - \frac{4V\left(1-4\bar{s}^2+4\tilde{s}\right)\left\langle[f(s)]^2\right\rangle}{(4\tilde{s}+1)^2}\right]^{-1}.$$

And for $\bar{s}$ we have an approximated Newton's iteration update candidate

$$\bar{s}_{new} = \bar{s} - \tilde{s}_{new}\left[\langle\exp v\rangle(\bar{s} - \langle m\rangle) + 4V\bar{s}\left(\frac{\left(\langle f(s)\rangle - \frac{M}{2V}\right)\langle f(s)\rangle}{2\tilde{s}+1} - \frac{\left\langle[f(s)]^2\right\rangle}{4\tilde{s}+1}\right)\right].$$

These candidates guarantee a direction, in which the cost function decreases locally. As long as the cost function is about to increase in value, the step size is halved. This guarantees the convergence to a stable point.

Appendix D. Example Where Point Estimates Fail

The following example illustrates what can go wrong with point estimates. Three dimensional data vectors $\mathbf{x}(t)$ are modelled with the linear factor analysis model $\mathbf{x}(t) = \mathbf{a}s(t) + \mathbf{n}(t)$, using a scalar source signal $s(t)$ and a Gaussian noise vector $\mathbf{n}(t)$ with zero mean and parameterised variance $p(n_k) = \mathcal{N}(0,\sigma_k^2)$. Here $\mathbf{a}$ is a three-dimensional weight vector.

The problem is to estimate $\mathbf{a}$ and $s(t)$ by maximising the likelihood $p(\mathbf{x}(t) \mid \mathbf{a}, s(t))$. The weight vector $\mathbf{a}$ might get a value $\mathbf{a} = [1\ \ 0\ \ 0]^T$, while the source can just copy the values of the first dimension of $\mathbf{x}(t)$, that is, $s(t) = x_1(t)$. When the reconstruction error or the noise term is evaluated: $\mathbf{n}(t) = \mathbf{x}(t) - \mathbf{a}s(t) = [0\ \ x_2(t)\ \ x_3(t)]^T$, one can see that problems will arise with the first variance parameter σ_1^2. The likelihood goes to infinity as σ_1^2 goes to zero. The same applies to the posterior density, since it is basically just the likelihood multiplied by a finite factor.

The found solution is completely useless and still, it is rated as infinitely good using point estimates. These problems are typical for models with estimates of the noise level or products. They can be sometimes avoided by fixing the noise level or using certain normalisations (Attias, 2001). When the noise model is nonstationary (see Section 5.1), the problem becomes even worse, since the infinite likelihood appears if the any of the variances goes to zero.

References

B. Anderson and J. Moore. *Optimal Filtering*. Prentice-Hall, Englewood Cliffs, NJ, 1979.

H. Attias. ICA, graphical models and variational methods. In S. Roberts and R. Everson, editors, *Independent Component Analysis: Principles and Practice*, pages 95–112. Cambridge University Press, 2001.

H. Attias. Independent factor analysis. *Neural Computation*, 11(4):803–851, 1999.

H. Attias. A variational Bayesian framework for graphical models. In T. Lee et al., editor, *Advances in Neural Information Processing Systems 12*, pages 209–215, Cambridge, 2000. MIT Press.

D. Barber and C. M. Bishop. Ensemble learning in Bayesian neural networks. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, pages 215–237. Springer, Berlin, 1998.

M. Beal. *Variational Algorithms for Approximate Bayesian Inference*. PhD thesis, University of London, UK, 2003.

M. Beal and Z. Ghahramani. The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian Statistics 7*, pages 453–464, 2003.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, 1995.

C. M. Bishop. Latent variable models. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 371–403. The MIT Press, Cambridge, MA, USA, 1999.

J.-F. Cardoso. Multidimensional independent component analysis. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'98)*, pages 1941–1944, Seattle, Washington, USA, May 12–15, 1998.

K. Chan, T.-W. Lee, and T. Sejnowski. Variational learning of clusters of undercomplete nonsymmetric independent components. In *Proc. Int. Conf. on Independent Component Analysis and Signal Separation (ICA2001)*, pages 492–497, San Diego, USA, 2001.

R. Choudrey, W. Penny, and S. Roberts. An ensemble learning approach to independent component analysis. In *Proc. of the IEEE Workshop on Neural Networks for Signal Processing, Sydney, Australia, December 2000*, pages 435–444. IEEE Press, 2000.

P. Dayan and R. Zemel. Competition and multiple cause models. *Neural Computation*, 7(3):565–579, 1995.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. of the Royal Statistical Society, Series B (Methodological)*, 39(1):1–38, 1977.

A. Doucet, N. de Freitas, and N. J. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer Verlag, 2001.

B. J. Frey and G. E. Hinton. Variational learning in nonlinear Gaussian belief networks. *Neural Computation*, 11(1):193–214, 1999.

A. Gelfand and A. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990.

A. Gelman, J. Carlin, H. Stern, and D. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC Press, Boca Raton, Florida, 1995.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

Z. Ghahramani and M. Beal. Propagation algorithms for variational Bayesian learning. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 507–513. The MIT Press, Cambridge, MA, USA, 2001.

Z. Ghahramani and G. E. Hinton. Hierarchical non-linear factor analysis and topographic maps. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems 10*, pages 486–492. The MIT Press, Cambridge, MA, USA, 1998.

Z. Ghahramani and S. Roweis. Learning nonlinear dynamical systems using an EM algorithm. In M. Kearns, S. Solla, and D. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 431–437. The MIT Press, Cambridge, MA, USA, 1999.

A. Gray, B. Fischer, J. Schumann, and W. Buntine. Automatic derivation of statistical algorithms: The EM family and beyond. In *Advances in Neural Information Processing Systems 15*, 2002. URL `http://www.hiit.fi/u/buntine/nips2002.htm`.

M. Harva. *Hierarchical Variance Models of Image Sequences*. Helsinki Univ. of Technology, Dept. of Computer Science and Eng., Espoo, Finland, March 2004. Master of Science (Dipl.Eng.) thesis. Available at `http://www.cis.hut.fi/mha`.

M. Harva and A. Kabán. Variational learning for rectified factor analysis. *Signal Processing*, 87(3): 509–527, 2007.

M. Harva, T. Raiko, A. Honkela, H. Valpola, and J. Karhunen. Bayes Blocks: An implementation of the variational Bayesian building blocks framework. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence, UAI 2005*, pages 259–266, Edinburgh, Scotland, July 2005.

S. Haykin, editor. *Kalman Filtering and Neural Networks*. Wiley, New York, 2001.

S. Haykin. *Neural Networks – A Comprehensive Foundation, 2nd ed*. Prentice-Hall, 1998.

G. E. Hinton and D. van Camp. Keeping neural networks simple by minimizing the description length of the weights. In *Proc. of the 6th Ann. ACM Conf. on Computational Learning Theory*, pages 5–13, Santa Cruz, CA, USA, 1993.

P. Højen-Sørensen, O. Winther, and L.K. Hansen. Mean-field approaches to independent component analysis. *Neural Computation*, 14(4):889–918, 2002.

A. Honkela. Speeding up cyclic update schemes by pattern searches. In *Proc. of the 9th Int. Conf. on Neural Information Processing (ICONIP'02)*, pages 512–516, Singapore, 2002.

A. Honkela and H. Valpola. Variational learning and bits-back coding: an information-theoretic view to Bayesian learning. *IEEE Transactions on Neural Networks*, 15(4):800–810, 2004.

A. Honkela and H. Valpola. Unsupervised variational Bayesian learning of nonlinear models. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 593–600. MIT Press, Cambridge, MA, USA, 2005.

A. Honkela, H. Valpola, and J. Karhunen. Accelerating cyclic update algorithms for parameter estimation by pattern searches. *Neural Processing Letters*, 17(2):191–203, 2003.

A. Honkela, S. Harmeling, L. Lundqvist, and H. Valpola. Using kernel PCA for initialisation of variational Bayesian nonlinear blind source separation method. In C. Puntonet and A. Prieto, editors, *Proc. of the Fifth Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA 2004)*, volume 3195 of *Lecture Notes in Computer Science*, pages 790–797, Granada, Spain, 2004. Springer-Verlag, Berlin.

A. Honkela, T. Östman, and R. Vigário. Empirical evidence of the linear nature of magnetoencephalograms. In *Proc. 13th European Symposium on Artificial Neural Networks (ESANN 2005)*, pages 285–290, Bruges, Belgium, 2005.

A. Hyvärinen and P. Hoyer. Emergence of phase and shift invariant features by decomposition of natural images into independent feature subspaces. *Neural Computation*, 12(7):1705–1720, 2000a.

A. Hyvärinen and P. Hoyer. Emergence of topography and complex cell properties from natural images using extensions of ICA. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 827–833. The MIT Press, Cambridge, MA, USA, 2000b.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. J. Wiley, 2001.

A. Ilin and H. Valpola. On the effect of the form of the posterior approximation in variational learning of ICA models. In *Proc. of the 4th Int. Symp. on Independent Component Analysis and Blind Signal Separation (ICA2003)*, pages 915–920, Nara, Japan, 2003.

A. Ilin, H. Valpola, and E. Oja. Nonlinear dynamical factor analysis for state change detection. *IEEE Trans. on Neural Networks*, 15(3):559–575, May 2003.

T. Jaakkola and M. I. Jordan. A variational approach to Bayesian logistic regression models and their extensions. In *Proceedings of the Sixth International Workshop on Artificial Intelligence and Statistics*, Fort Lauderdale, Florida, January 1997.

M. I. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. An introduction to variational methods for graphical models. In M. Jordan, editor, *Learning in Graphical Models*, pages 105–161. The MIT Press, Cambridge, MA, USA, 1999.

T. Kohonen. *Self-Organizing Maps*. Springer, 3rd, extended edition, 2001.

T. Kohonen, S. Kaski, and H. Lappalainen. Self-organized formation of various invariant-feature filters in the Adaptive-Subspace SOM. *Neural Computation*, 9(6):1321–1344, 1997.

H. Lappalainen and A. Honkela. Bayesian nonlinear independent component analysis by multi-layer perceptrons. In M. Girolami, editor, *Advances in Independent Component Analysis*, pages 93–121. Springer-Verlag, Berlin, 2000.

H. Lappalainen and J. Miskin. Ensemble learning. In M. Girolami, editor, *Advances in Independent Component Analysis*, pages 75–92. Springer-Verlag, Berlin, 2000.

D. J. C. MacKay. Local minima, symmetry-breaking, and model pruning in variational free energy minimization. Available at `http://www.inference.phy.cam.ac.uk/mackay/`, 2001.

D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

D. J. C. MacKay. Developments in probabilistic modelling with neural networks – ensemble learning. In *Neural Networks: Artificial Intelligence and Industrial Applications. Proc. of the 3rd Annual Symposium on Neural Networks*, pages 191–198, 1995.

D. J. C. MacKay. Ensemble learning for hidden Markov models. Available at `http://wol.ra.phy.cam.ac.uk/mackay/`, 1997.

T. Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence, UAI 2001*, pages 362–369, Seattle, Washington, USA, 2001.

J. Miskin and D. J. C. MacKay. Ensemble learning for blind source separation. In S. Roberts and R. Everson, editors, *Independent Component Analysis: Principles and Practice*, pages 209–233. Cambridge University Press, 2001.

K. Murphy. The Bayes net toolbox for Matlab. *Computing Science and Statistics*, 33:331–350, 2001.

K. Murphy. A variational approximation for Bayesian networks with discrete and continuous latent variables. In *Proc. of the 15th Annual Conf. on Uncertainty in Artificial Intelligence (UAI–99)*, pages 457–466, Stockholm, Sweden, 1999.

R. Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, April 2001.

R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. The MIT Press, Cambridge, MA, USA, 1999.

L. Nolan, M. Harva, A. Kabán, and S. Raychaudhury. A data-driven Bayesian approach for finding young stellar populations in early-type galaxies from their UV-optical spectra. *Monthly Notices of the Royal Astronomical Society*, 366(1):321–338, 2006.

H.-J. Park and T-W. Lee. A hierarchical ICA method for unsupervised learning of nonlinear dependencies in natural images. In C. Puntonet and A. Prieto, editors, *Proc. of the 5th Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA2004)*, pages 1253–1261, Granada, Spain, 2004.

L. Parra, C. Spence, and P. Sajda. Higher-order statistical properties arising from the non-stationarity of natural signals. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 786–792. The MIT Press, Cambridge, MA, USA, 2001.

J. Pearl, editor. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Francisco, California, 1988.

D.-T. Pham and J.-F. Cardoso. Blind separation of instantaneous mixtures of nonstationary sources. *IEEE Trans. on Signal Processing*, 49(9):1837–1848, 2001.

T. Raiko. Partially observed values. In *Proc. Int. Joint Conf. on Neural Networks (IJCNN'04)*, pages 2825–2830, Budapest, Hungary, 2004.

T. Raiko. Nonlinear relational Markov networks with an application to the game of Go. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN 2005)*, pages 989–996, Warsaw, Poland, September 2005.

T. Raiko and M. Tornio. Learning nonlinear state-space models for control. In *Proc. Int. Joint Conf. on Neural Networks (IJCNN'05)*, pages 815–820, Montreal, Canada, 2005.

T. Raiko, H. Valpola, T. Östman, and J. Karhunen. Missing values in hierarchical nonlinear factor analysis. In *Proc. of the Int. Conf. on Artificial Neural Networks and Neural Information Processing (ICANN/ICONIP 2003)*, pages 185–189, Istanbul, Turkey, 2003.

T. Raiko, M. Tornio, A. Honkela, and J. Karhunen. State inference in variational Bayesian nonlinear state-space models. In *Proceedings of the 6th International Conference on Independent Component Analysis and Blind Source Separation (ICA 2006)*, pages 222–229, Charleston, South Carolina, USA, March 2006.

S. Roberts and R. Everson, editors. *Independent Component Analysis: Principles and Practice*. Cambridge Univ. Press, 2001.

S. Roberts, E. Roussos, and R. Choudrey. Hierarchy, priors and wavelets: structure and signal modelling using ICA. *Signal Processing*, 84(2):283–297, February 2004.

D. Rowe. *Multivariate Bayesian Statistics: Models for Source Separation and Signal Unmixing*. Chapman & Hall/CRC, Medical College of Wisconsin, 2003.

G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

D. J. Spiegelhalter, A. Thomas, N. G. Best, and W. R. Gilks. BUGS: Bayesian inference using Gibbs sampling, version 0.50. Available at `http://www.mrc-bsu.cam.ac.uk/bugs/`, 1995.

T. Swartz and M. Evans. *Approximating Integrals Via Monte Carlo and Deterministic Methods*. Oxford University Press, 2000.

L. Tierney and J.B. Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81(393):82–86, 1986.

H. Valpola and J. Karhunen. An unsupervised ensemble learning method for nonlinear dynamic state-space models. *Neural Computation*, 14(11):2647–2692, 2002.

H. Valpola, T. Raiko, and J. Karhunen. Building blocks for hierarchical latent variable models. In *Proc. 3rd Int. Conf. on Independent Component Analysis and Signal Separation (ICA2001)*, pages 710–715, San Diego, USA, 2001.

H. Valpola, A. Honkela, and J. Karhunen. An ensemble learning approach to nonlinear dynamic blind source separation using state-space models. In *Proc. Int. Joint Conf. on Neural Networks (IJCNN'02)*, pages 460–465, Honolulu, Hawaii, USA, 2002.

H. Valpola, A. Honkela, M. Harva, A. Ilin, T. Raiko, and T. Östman. Bayes Blocks software library, 2003a. Available at `http://www.cis.hut.fi/projects/bayes/software/`.

H. Valpola, E. Oja, A. Ilin, A. Honkela, and J. Karhunen. Nonlinear blind source separation by variational Bayesian learning. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E86-A(3):532–541, 2003b.

H. Valpola, T. Östman, and J. Karhunen. Nonlinear independent factor analysis by hierarchical models. In *Proc. 4th Int. Symp. on Independent Component Analysis and Blind Signal Separation (ICA2003)*, pages 257–262, Nara, Japan, 2003c.

H. Valpola, M. Harva, and J. Karhunen. Hierarchical models of variance sources. *Signal Processing*, 84(2):267–282, 2004.

J. H. van Hateren and D. L. Ruderman. Independent component analysis of natural image sequences yields spatio-temporal filters similar to simple cells in primary visual cortex. *Proceedings of the Royal Society of London B*, 265(1412):2315–2320, 1998.

J. Vesanto, J. Himberg, E. Alhoniemi, and J. Parhankangas. Self-organizing map in Matlab: the SOM toolbox. In *Proceedings of the Matlab DSP Conference*, pages 35–40, Espoo, Finland, November 1999. Available at `http://www.cis.hut.fi/projects/somtoolbox/`.

C. Wallace. Classification by minimum-message-length inference. In S. G. Aki, F. Fiala, and W. W. Koczkodaj, editors, *Advances in Computing and Information – ICCI '90*, volume 468 of *Lecture Notes in Computer Science*, pages 72–81. Springer, Berlin, 1990.

S. Waterhouse, D. MacKay, and T. Robinson. Bayesian methods for mixtures of experts. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 351–357. The MIT Press, 1996.

J. Winn and C. M. Bishop. Variational message passing. *Journal of Machine Learning Research*, 6: 661–694, April 2005.

Journal of Machine Learning Research 8 (2007) 203-226 Submitted 10/05; Revised 9/06; Published 2/07

A Probabilistic Analysis of EM for Mixtures of Separated, Spherical Gaussians

Sanjoy Dasgupta DASGUPTA@CS.UCSD.EDU
University of California, San Diego
9500 Gilman Drive #0404
La Jolla, CA 92093-0404

Leonard Schulman SCHULMAN@CALTECH.EDU
California Institute of Technology
1200 E. California Blvd., MC 256-80
Pasadena, CA 91125

Editor: Leslie Pack Kaelbling

Abstract

We show that, given data from a mixture of k well-separated spherical Gaussians in $\mathbb{R}^d$, a simple two-round variant of EM will, with high probability, learn the parameters of the Gaussians to near-optimal precision, if the dimension is high ($d \gg \ln k$). We relate this to previous theoretical and empirical work on the EM algorithm.

Keywords: expectation maximization, mixtures of Gaussians, clustering, unsupervised learning, probabilistic analysis

1. Introduction

At present the *expectation-maximization* algorithm (Dempster, Laird, and Rubin, 1977; Wu, 1983) is the method of choice for learning mixtures of Gaussians. A series of theoretical and experimental studies over the past three decades have contributed to the collective intuition about this algorithm. We will reinterpret some of these results in the context of a new performance guarantee.

In brief, EM is a hillclimbing algorithm which starts with some initial estimate of the Gaussian mixture and then repeatedly changes this estimate so as to improve its likelihood, until it finally converges to a local optimum of the search space. It is well known to practitioners that the quality of the output can be influenced significantly by the manner in which EM is initialized. Another source of variation is that in practice it is quite common to add or remove Gaussians during the search process, according to various intuitively-motivated criteria.

Given the enormous importance of Gaussian mixture models in applied statistics and machine learning, it is reasonable to wonder how good the EM algorithm is. Among other things, a rigorous analysis of its performance might shed light on some of the unresolved issues in its implementation—for instance, initialization. It is in this spirit that we approach our analysis of EM.

A common form of theoretical study is *worst-case analysis*, which in this case would attempt to bound how far EM can deviate from the optimal log-likelihood, or perhaps the optimal set of mixture parameters. Such an analysis turns out to be trivial, because—as is well known—EM's output can be arbitrarily far from optimal for either of the two criteria above (we will see such an example in Section 2.3). Thus, this line of reasoning does not appear to yield any interesting insights into the algorithm's behavior.

In this paper, we perform what might be called a *best-case analysis*. We assume that the data are the best EM could possibly hope for: i.i.d. samples from a mixture of spherical Gaussians in $\mathbb{R}^d$ which are well separated from one another. At first glance, it seems that we are once again in danger of getting a trivial result, namely that EM will succeed without a hiccup. But this is not the case. In fact, we will see that even in this extremely optimistic setting, many common ways of initializing, and subsequently running, EM will make it fail dramatically. On the other hand, if EM is run in a particular manner which we specify, then within just two rounds, it will identify the correct mixture parameters with near-perfect accuracy.

If the data is expected to have k clusters, it is common for practitioners to start EM with more than k centers, and to later prune some of these extra centers. We present a simple example to demonstrate exactly why this is necessary, and obtain an expression for the number of initial centers which should be used: at least $\frac{1}{w_{min}} \ln k$, where w_{min} is a lower bound on the smallest mixing weight. The typical method of pruning is to remove Gaussian-estimates with very low mixing weight (known as *starved clusters*). Our theoretical analysis shows that this is not enough, that there is another type of Gaussian-estimate, easy to detect, which also needs to be pruned. Specifically, it is possible (and frequently occurs in simulations) that two of EM's Gaussian-estimates share the same cluster, each with relatively high mixing weight. We present a very simple, provably correct method of detecting this situation and correcting it.

It is widely recognized that a crucial issue in the performance of EM is the choice of initial parameters. For the means, we use the popular technique of selecting them randomly from the data set. This is shown to be adequate for the performance guarantee we derive. Our analysis also makes it clear that it is vitally important to pick good initial estimates of the covariances, a subject which has received somewhat less attention. We use an initializer whose origin we are unable to trace but which is mentioned in Bishop's text (1995).

Our analysis is focused on the case when the data are high-dimensional ($d \gg \ln k$), and it brings out some interesting qualitative differences from the low-dimensional case. In particular, it is common to think of EM as assigning "soft" cluster memberships in which each data point is not definitively assigned to a single cluster but, rather, is split between clusters according to its relative proximities to them. Moreover, this is sometimes quoted as a source of EM's effectiveness—that these soft memberships allow cluster boundaries to shift in a smooth and stable manner. In the optimistic high-dimensional scenario we analyze, the behavior is quite different, in that cluster memberships are effectively always "hard". This is because the distances are so large that for any two clusters, there is only a small part of the space in which there is any ambiguity of assignment, and it is unlikely that data points would lie in these zones. Moreover, the phenomenon of smooth transitions between clusterings is nonexistent. Instead, EM quickly snaps into one of several trajectories (loosely defined), and thereafter heads to a nearby local optimum. Initialization is of supreme importance—once EM has snapped into the wrong trajectory, all is lost.

1.1 Relation to Previous Work on EM

A standard criticism of EM is that it converges slowly. Simulations performed by Redner and Walker (1984), and others, demonstrate this decisively for one-dimensional mixtures of two Gaussians. It is also known that given data from a mixture of Gaussians, when EM gets close to the true solution, it exhibits *first-order convergence* (Titterington, Smith, and Makov, 1985). Roughly speaking, the idea is this: given m data points from a mixture with parameters (means, covariances, and mixing weights) θ^*, where m is very large, the likelihood has a local maximum at some set of parameters θ^m close to θ^*. Let $\theta^{\langle t\rangle}$ denote EM's parameter-estimates at time t. It can be shown (cf. Taylor expansion) that when $\theta^{\langle t\rangle}$ is near θ^m,

$$\|\theta^{\langle t+1\rangle} - \theta^m\| \leq \lambda \cdot \|\theta^{\langle t\rangle} - \theta^m\|,$$

where $\lambda \in [0,1)$ and $\|\cdot\|$ is some norm.[1] If the Gaussians are closely packed then λ is close to one; if they are very far from one another then λ is close to zero.

Xu and Jordan (1995) present theoretical results which mitigate some of the pessimism of first-order convergence, particularly in the case of well-separated mixtures, and they note that moreover near-optimal log-likelihood is typically reached in just a few iterations. We also argue in favor of EM, but in a different way. We ask, how close does $\theta^{\langle t\rangle}$ have to be to θ^m for slow convergence to hold? Let $D(\theta_1,\theta_2)$ denote the maximum Euclidean distance between the respective means of θ_1 and θ_2. For one-dimensional data, it can be seen quite easily from canonical experiments (Redner and Walker, 1984) that convergence is slow even if $D(\theta^{\langle t\rangle},\theta^*)$ is large. However, our results suggest that this no longer holds in higher dimension. For reasonably well-separated spherical Gaussians in $\mathbb{R}^d$ (where *separation* is defined precisely in the next section), convergence is very fast until $D(\theta^{\langle t\rangle},\theta^*) \approx e^{-\Omega(d)}$. In fact, we can make EM attain this accuracy in just two rounds. The error $e^{-\Omega(d)}$ is so miniscule for large d that subsequent improvements are not especially important.

At a high level, previous analyses of EM have typically adopted an *optimization-based* viewpoint: they have studied EM by studying the objective function (log-likelihood) that it is ostensibly optimizing. A typical tool in this kind of analysis is to perform a Taylor expansion of the log-likelihood in the vicinity of a local optimum, assuming the data are i.i.d. draws from a mixture of Gaussians, and to thereby get insight into the speed at which EM is likely to move when it gets close to this optimum. A major drawback of this approach is that it only addresses what happens *close to convergence*. It cannot, for instance, give intuition about how to initialize EM, or about whether a local optimum of high quality is attained.

In contrast, we perform a *probabilistic* analysis. We also assume that the data are i.i.d. draws from a mixture of Gaussians, but we focus upon the actual algorithm and ignore the likelihood function altogether. We ask, what will the algorithm do in step one, with high probability over the choice of data? In step two? And so on. This enables us to address issues of initialization and global optimality.

1.2 Results

Performance guarantees for clustering will inevitably involve some notion of the *separation* between different clusters. There are at least two natural ways of defining this. Take for simplicity the case of two d-dimensional Gaussians $N(\mu_1,I_d)$ and $N(\mu_2,I_d)$. If each coordinate (attribute) provides a little bit of discriminative information between the two clusters, then on each coordinate μ_1 and μ_2 differ by at least some small amount, say δ. The L_2 distance between μ_1 and μ_2 is then at least $\delta\sqrt{d}$. As further attributes are added, the distance between the centers grows, and the two clusters become more clearly distinguishable from one another. This is the usual rationale for using high-dimensional data: the higher the dimension, the easier (in an information-theoretic sense) clustering

1. This might not seem so bad, but contrast it with *second-order convergence*, in which $\|\theta^{\langle t+1\rangle} - \theta^m\| \leq \lambda \cdot \|\theta^{\langle t\rangle} - \theta^m\|^2$.

should be. The only problem then, is whether there are algorithms which can efficiently exploit the tradeoff between this high information content and the curse of dimensionality. This viewpoint suggests that the Euclidean distance between the centers of d-dimensional clusters can reasonably be measured in units of $\sqrt{d}$, and that it is most important to develop algorithms which work well under the assumption that this distance is some constant times $\sqrt{d}$. On the other hand, it should be pointed out that if $\|\mu_1 - \mu_2\| = \delta\sqrt{d}$ for some constant $\delta > 0$, then for large d the overlap in probability mass between the two Gaussians is miniscule, exponentially small in d. Therefore, it should not only be interesting but also possible to develop algorithms which work well when the L_2 distance between centers of clusters is some constant, regardless of the dimension (as opposed to a constant times $\sqrt{d}$).

Where does EM fall in this spectrum of separation? It lies somewhere in between: we show that it works well when the distance between d-dimensional clusters is bigger than $d^{1/4}$.

Our central performance guarantee requires that the clusters actually look spherical-Gaussian, more specifically that the data points are drawn i.i.d. from some (unknown) mixture of spherical Gaussians, which could potentially have different variances. We show that if the clusters are reasonably well-separated (in the sense we just defined), and if the dimension $d \gg \ln k$ then only two rounds of EM are required to learn the mixture to within near-optimal precision, with high probability $1 - k^{-\Omega(1)}$. Our measure of accuracy is the function $D(\cdot,\cdot)$ introduced above. The precise statement of the theorem can be found in the next section, and applies not only to EM but also to other similar schemes, including for instance some of the variants of EM and k-means introduced by Kearns, Mansour, and Ng (1997).

Several recent papers have suggested alternative algorithms for learning the parameters of a mixture of well-separated Gaussians (or other distributions with similar concentration properties), given data drawn i.i.d. from that distribution. The first in this series, by Dasgupta (1999), requires the Gaussians to be "sphere-like" and separated by a distance of $\Omega(\sqrt{d})$. Arora and Kannan (2004) handle more general Gaussians, and reduce the separation requirement to $\Omega(d^{1/4})$ in the spherical case (as in this paper). Vempala and Wang (2004) use spectral projection to bring the separation constraint for spherical Gaussians down to just $\Omega((k \log d)^{1/4})$, where k is the number of clusters. Vempala, Kannan, and Salmasian (2005) and Achlioptas and McSherry (2005) give extensions of these latter results to ellipsoidal clusters. The last three results are especially relevant to the current paper because the amount of intercluster separation we require can likely be substantially reduced if spectral projection is used as a preprocessing step.

In the final section of the paper, we discuss a crucial issue: what features of our main assumption (that the clusters are high-dimensional Gaussians) make our guarantees for EM possible? This assumption is also the basis of the other theoretical results mentioned above, but can real data sets reasonably be expected to satisfy it? If not, in what way can it usefully be relaxed?

2. Statement of Results

To motivate our model, we start by examining some properties of high-dimensional Gaussians.

2.1 High-dimensional Gaussians

A spherical Gaussian $N(\mu, \sigma^2 I_d)$ assigns to point $x \in \mathbb{R}^d$ the density

$$p(x) = \frac{1}{(2\pi)^{d/2}\sigma^d} \exp\left(-\frac{\|x-\mu\|^2}{2\sigma^2}\right),$$

$\|\cdot\|$ being Euclidean distance. If $X = (X_1, \ldots, X_d)$ is randomly chosen from $N(0, \sigma^2 I_d)$ then its coordinates are i.i.d. $N(0, \sigma^2)$ random variables. Each coordinate has expected squared value σ^2 so $\mathbb{E}\|X\|^2 = \mathbb{E}(X_1^2 + \cdots + X_d^2) = \sigma^2 d$. It then follows by a large deviation bound that $\|X\|^2$ will be tightly concentrated around $\sigma^2 d$:

$$\mathbb{P}(|\|X\|^2 - \sigma^2 d| > \varepsilon\sigma^2 d) \leq e^{-d\varepsilon^2/8}.$$

This bound and others like it will be proved in Section 3. It means that almost the entire probability mass of $N(0, \sigma^2 I_d)$ lies in a thin shell at a radius of about $\sigma\sqrt{d}$ from the origin. The density of the Gaussian is highest at the origin; however, the surface area at distance r from the origin, $0 \leq r \leq \sigma\sqrt{d}$, increases faster than the density at distance r decreases (Bishop, 1995, exercise 1.4).

It is natural therefore to think of a Gaussian $N(\mu, \sigma^2 I_d)$ as having *radius* $\sigma\sqrt{d}$. We say two Gaussians $N(\mu_1, \sigma_1^2 I_d), N(\mu_2, \sigma_2^2 I_d)$ in $\mathbb{R}^d$ are *c-separated* if

$$\|\mu_1 - \mu_2\| \geq c \max\{\sigma_1, \sigma_2\}\sqrt{d},$$

that is, if they are c radii apart. A mixture of Gaussians is c-separated if the Gaussians in it are pairwise c-separated. In general we will let c_{ij} denote the separation between the i^{th} and j^{th} Gaussians, and $c = \min_{i \neq j} c_{ij}$. We can reasonably expect that the difficulty of learning a mixture of Gaussians increases as c decreases.

A 2-separated mixture of Gaussians contains clusters with almost no overlap. For large d, this is true even of a $\frac{1}{100}$-separated mixture, because for instance, the two balls $B(0, \sqrt{d})$ and $B(\frac{1}{100}\sqrt{d}, \sqrt{d})$ in $\mathbb{R}^d$ share only a tiny fraction of their volume. One useful way of thinking about a pair of c-separated Gaussians is to imagine that on each coordinate their means differ by c. If c is small, then the projection of the mixture onto any one coordinate will look unimodal. This might also be true of a projection onto a few coordinates. But for large d, when all coordinates are considered together, the distribution will cease to be unimodal. This is precisely the reason for using high-dimensional data.

What values of c can be expected of real-world data sets? This will vary from case to case. As an example, we analyzed a canonical data set consisting of handwritten digits collected by USPS. Each digit is represented as a vector in $[-1, 1]^{256}$. We fit a mixture of ten (non-spherical) Gaussians to this data set, by doing each digit separately, and found that it was 0.63-separated.

2.2 The EM algorithm

A mixture of k spherical Gaussians in $\mathbb{R}^d$ is specified by a set of mixing weights w_i (which sum to one and represent the proportions in which the various Gaussians are present) and by the individual Gaussian means $\mu_i \in \mathbb{R}^d$ and variances σ_i^2.

Given a data set $S \in \mathbb{R}^d$, the EM algorithm works by first choosing starting values $w_i^{\langle 0 \rangle}, \mu_i^{\langle 0 \rangle}, \sigma_i^{\langle 0 \rangle}$ for the parameters, and then updating them iteratively according to the following two steps (at time t).

E step Let $\tau_i \sim N(\mu_i^{\langle t\rangle}, \sigma_i^{\langle t\rangle 2} I_d)$ denote the density of the i^{th} Gaussian-estimate. For each data point $x \in S$, and each $1 \le i \le k$, compute

$$p_i^{\langle t+1\rangle}(x) = \frac{w_i^{\langle t\rangle}\tau_i(x)}{\sum_j w_j^{\langle t\rangle}\tau_j(x)},$$

the conditional probability that x comes from the i^{th} Gaussian with respect to the current estimated parameters.

M step Now update the various parameters in an intuitive way. Let m be the size of S.

$$\begin{aligned} w_i^{\langle t+1\rangle} &= \frac{1}{m}\sum_{x\in S} p_i^{\langle t+1\rangle}(x), \\ \mu_i^{\langle t+1\rangle} &= \frac{1}{mw_i^{\langle t+1\rangle}}\sum_{x\in S} x\, p_i^{\langle t+1\rangle}(x), \\ \sigma_i^{\langle t+1\rangle 2} &= \frac{1}{mw_i^{\langle t+1\rangle} d}\sum_{x\in S} \|x-\mu_i^{\langle t+1\rangle}\|^2\, p_i^{\langle t+1\rangle}(x). \end{aligned}$$

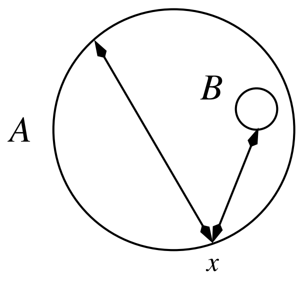
2.3 The Main Issues

We now give a high-level description of some fundamental issues that arise in our analysis of EM, and that dictate our key design decisions.

2.3.1 TIGHT CONCENTRATION OF INTERPOINT DISTANCES

In a high-dimensional space $\mathbb{R}^d$, the distances between data points—whether sampled from the same Gaussian or from different Gaussians—are tightly concentrated around their expected values. In particular, if the Gaussians happen to have the same variance $\sigma^2 I_d$, and if the distances between their centers are $\gg \sigma d^{1/4}$, then the chance that two points from different Gaussians are closer together than two points from the same Gaussian, is tiny, $e^{-\Omega(poly(d))}$. Therefore, an examination of interpoint distances is enough to almost perfectly cluster the data. A variety of different algorithms will work well under these circumstances, and EM is no exception.

What if the Gaussians have different variances σ_i? Once again, interpoint distances are close to their expected values, but now a new complication is introduced. If a small-variance cluster B is close to the center of a larger-variance cluster A, then it is quite possible for points $x \in A$ to be closer to all of B than to any other point in A:



We expressly rule out this case by requiring the separation between any two clusters i and j to satisfy

$$\|\mu_i - \mu_j\|^2 \ge |\sigma_i^2 - \sigma_j^2| d.$$

2.3.2 INITIALIZATION

Suppose the true number of Gaussians, k, is known. Let S denote the entire data set, with S_i being the points drawn from the i^{th} true Gaussian $N(\mu_i, \sigma^2 I_d)$. A common way to initialize EM is to pick l data points at random from S, and to use these as initial *center-estimates* $\mu_i^{\langle 0 \rangle}$. How large should l be? It turns out that if these l points include at least one point from each S_i, then EM can be made to perform well. This suggests $l = \Omega(k \ln k)$. Conversely, if the initial centers miss some S_i, then EM might perform poorly.

Here is a concrete example (Figure 1). Let d denote some high dimension, and place the k true Gaussians $N(\mu_1, I_d), \ldots, N(\mu_k, I_d)$ side by side in a line, leaving a distance of at least $3\sqrt{d}$ between consecutive Gaussians. Assign them equal mixing weights. As before let S_i be the data points from the i^{th} Gaussian, and choose EM's initial center-estimates from the data. Suppose the initial centers contain nothing from S_1, one point from S_2, and at least one point from S_3. The probability of this event is at least some constant. Then no matter how long EM is run, it will assign just one Gaussian-estimate to the first two true Gaussians. In the first round of EM, the point from S_2 (call it $\mu_1^{\langle 0 \rangle}$) will move between μ_1 and μ_2. It will stay there, right between the two true centers. None of the other center-estimates $\mu_i^{\langle t \rangle}$ will ever come closer to μ_2; their distance from it is so large that their influence is overwhelmed by that of $\mu_1^{\langle t \rangle}$. This argument can be formalized easily using the large deviation bounds that we will introduce in the next section.

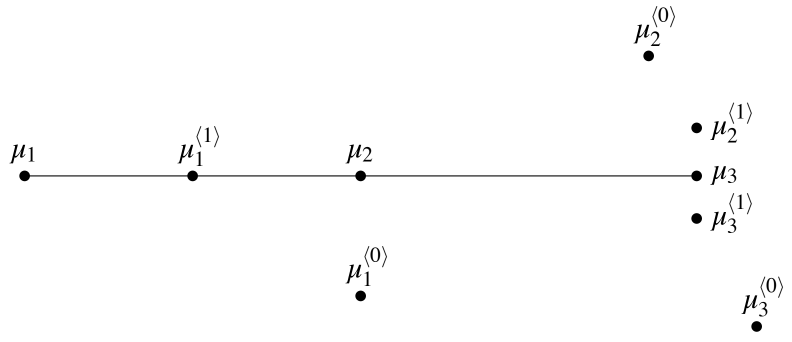

$\mu_2^{\langle 0 \rangle}$
$\mu_2^{\langle 1 \rangle}$
μ_1
$\mu_1^{\langle 1 \rangle}$
μ_2
μ_3
$\mu_3^{\langle 1 \rangle}$
$\mu_1^{\langle 0 \rangle}$
$\mu_3^{\langle 0 \rangle}$

Figure 1: For this mixture, the positions of the center-estimates do not move much after the first step of EM.

How about the initial choice of variance? In the case when the Gaussians have the same spherical covariance, this is not all that important, except that a huge overestimate might cause slower convergence. In the case when the Gaussians have different variances, however, the initial estimates are crucially important, and so we will use a fairly precise estimator, a variant of which is mentioned in Bishop's text (1995).

2.3.3 AFTER THE FIRST ROUND OF EM

After one round of EM, the center-estimates are pruned to leave exactly one per true Gaussian. This is accomplished in a simple manner. First, remove any center-estimates with very low mixing weight (this is often called "cluster starvation"). Any remaining center-estimate (originally chosen, say,

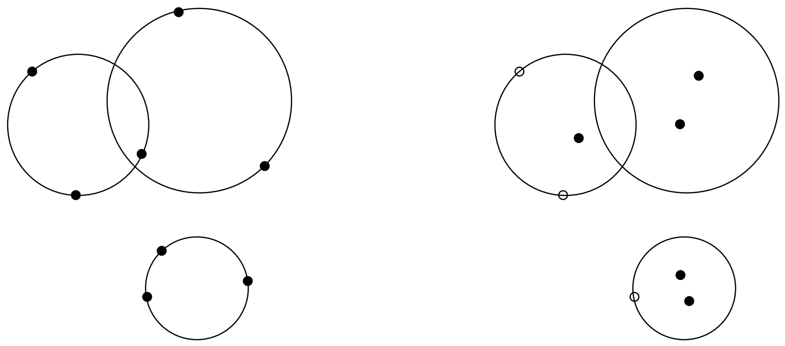


Figure 2: The large circles are true clusters, while the dots (solid or hollow) are EM's center-estimates. *Left:* after initialization, we have at least one center-estimate per true cluster. *Right:* After the first round of EM, each center-estimate has either been "starved" (shown as hollow) or has moved closer to the corresponding true center.

from S_i) has relatively high mixing weight, and we can show that as a result of the first EM iteration, it will have moved close to μ_i (Figure 2). A trivial clustering heuristic, due to Gonzalez (1985), is then good enough to select one center-estimate near each μ_i.

With exactly one center-estimate per (true) Gaussian, a second iteration of EM will accurately retrieve the means, variances, and mixing weights. In fact the clustering of the data (the fractional labels assigned by EM) will be almost perfect, that is to say, each fractional label will be close to zero or one, and will in almost all cases correctly identify the generating Gaussian. Therefore further iterations will not help much: these additional iterations will move the center-estimates around by at most $e^{-\Omega(d)}$.

2.4 A Two-round Variant of EM

Here is a summary of the modified algorithm, given m data points in $\mathbb{R}^d$ which have been generated by a mixture of k Gaussians. The value of l will be specified later; for the time being it can be thought of as $O(k \ln k)$.

Initialization Pick l data points at random as starting estimates $\mu_i^{\langle 0\rangle}$ for the Gaussian centers. Assign them identical mixing weights $w_i^{\langle 0\rangle} = \frac{1}{l}$. For initial estimates of the variances use

$$\sigma_i^{\langle 0\rangle 2} = \frac{1}{2d} \min_{j\neq i} \|\mu_i^{\langle 0\rangle} - \mu_j^{\langle 0\rangle}\|^2.$$

EM Run one round of EM. This yields modified estimates $w_i^{\langle 1\rangle}, \mu_i^{\langle 1\rangle}, \sigma_i^{\langle 1\rangle}$.

Pruning Remove all center-estimates whose mixing weights are below $w_T = \frac{1}{4l}$. Then, prune the remaining center-estimates down to just k by using the following adaptation of an algorithm of Gonzalez (1985):

- Compute distances between center-estimates:

$$d(\mu_i^{\langle 1\rangle},\mu_j^{\langle 1\rangle}) = \frac{\|\mu_i^{\langle 1\rangle} - \mu_j^{\langle 1\rangle}\|}{\sigma_i^{\langle 1\rangle} + \sigma_j^{\langle 1\rangle}}.$$

- Choose one of these centers arbitrarily.
- Pick the remaining $k-1$ iteratively as follows: pick the center farthest from the ones picked so far. (The distance from a point x to a set S is $\min_{y\in S} d(x,y)$.)

Call the resulting center-estimates $\tilde{\mu}_i^{\langle 1\rangle}$ (where $1 \le i \le k$), and let $\tilde{\sigma}_i^{\langle 1\rangle 2}$ be the corresponding variances. Set the mixing weights to $\tilde{w}_i^{\langle 1\rangle} = \frac{1}{k}$.

EM Run one more step of EM, starting at the $\{\tilde{w}_i^{\langle 1\rangle}, \tilde{\mu}_i^{\langle 1\rangle}, \tilde{\sigma}_i^{\langle 1\rangle}\}$ parameters and yielding the final estimates $w_i^{\langle 2\rangle}, \mu_i^{\langle 2\rangle}, \sigma_i^{\langle 2\rangle}$.

2.5 The Main Result

Now that the notation and algorithm have been introduced, we can state the main theorem.

Theorem 1 *Say m data points are generated from a mixture of k Gaussians in* $\mathbb{R}^d$,

$$w_1 N(\mu_1, \sigma_1^2 I_d) + \cdots + w_k N(\mu_k, \sigma_k^2 I_d),$$

where the intercenter distances satisfy the inequality $\|\mu_i - \mu_j\|^2 \ge |\sigma_i^2 - \sigma_j^2| d$.

Define c_{ij} *to be the separation between the* i^{th} *and* j^{th} *Gaussians—that is,* $\|\mu_i - \mu_j\| = c_{ij}\max(\sigma_i,\sigma_j)\sqrt{d}$ *— and* $c = \min_{i\neq j} c_{ij}$ *to be the overall separation. Let* S_i *denote the points from the* i^{th} *Gaussian, and let* $w_{min} = \min_i w_i$*. For any* $\delta > 0$ *and* $\varepsilon > 0$*, if*

1. *parameter* $l = \Omega(\frac{1}{w_{min}} \ln \frac{1}{\delta w_{min}})$,
2. *dimension* $d = \Omega(\max(1, c^{-4}) \ln \frac{\max(1,c^{-4})}{\delta w_{min}})$,
3. *number of samples* $m = \Omega(l \max(1, c^{-2}))$,
4. *separation* $c^2 d = \Omega(\ln \frac{1}{\varepsilon w_{min}})$,

then with probability at least $1-\delta$*, the variant of EM described above will produce final center-estimates which (appropriately permuted) satisfy*

$$\|\mu_i^{\langle 2\rangle} - \mu_i\| \;\le\; \|\text{mean}(S_i) - \mu_i\| + \varepsilon\sigma_i\sqrt{d}.$$

This theorem will be proved over the remainder of the paper, but a few words about it are in order at this stage. First of all, the constants which have been left out of the theorem statement are given in Section 4. Second, the best that can be hoped is that $\mu_i^{\langle 2\rangle} = \text{mean}(S_i)$; therefore, the final error bound on the center-estimates becomes very close to optimal as $c^2 d$ increases. Third, similarly strong bounds can be given for the final mixing weights and variances; see Theorem 17. Finally, notice that the bounds require $c \gg d^{-1/4}$; in other words, the distance between the centers of Gaussians i and j must be $\gg \max(\sigma_i,\sigma_j) d^{1/4}$.

d	dimension
k	true number of clusters
w_i, μ_i, σ_i^2	true mixture parameters
σ_{ij}	shorthand for $\max(\sigma_i, \sigma_j)$
w_{min}	lower bound on the mixing weights: $w_i \geq w_{min}$
l	number of clusters with which EM is started; $l > k$
$w_i^{\langle t\rangle}, \mu_i^{\langle t\rangle}, \sigma_i^{\langle t\rangle 2}$	EM's parameter-estimates at time t
$p_i^{\langle t\rangle}(x)$	EM's soft-assignment (to point x, from cluster i) at time t
w_T	threshold used to prune "starved clusters": $w_T = 1/4l$
c_{ij}	separation between Gaussians i and j; $\|\mu_i - \mu_j\| = c_{ij}\sigma_{ij}\sqrt{d}$
c	overall separation, $c = \min_{i\neq j} c_{ij}$
S	data points
S_i	data points sampled from the i^{th} Gaussian
m	number of data points
ε_o	concentration of interpoint distances and dot products
C_i	center-estimates from S_i which survived the first round: $C_i = \{\mu_{i'}^{\langle 1\rangle} : \mu_{i'}^{\langle 0\rangle} \in S_i, w_{i'}^{\langle 1\rangle} \geq w_T\}$
$d(\mu_i^{\langle 1\rangle}, \mu_j^{\langle 1\rangle})$	weighted distance between centers, used by pruning procedure

Figure 3: Notation.

3. Concentration Properties of Gaussian Samples

Our analysis hinges crucially upon concentration effects: specifically, that interpoint distances and means of subsets of points are likely to be close to their expected values.

The most basic such property is that the squared length of a point drawn from a high-dimensional Gaussian is tightly concentrated. The proof of this well-known fact is repeated here for easy reference.

Lemma 2 *Pick X from the distribution $N(0, I_d)$.*

(a) (Very large deviations) For any $\lambda \geq 1$, we have $\mathbf{P}(\|X\|^2 \geq \lambda d) \leq (e^{\lambda - 1 - \ln\lambda})^{-d/2}$.

(b) (Modest deviations) For any $\varepsilon \in (0,1)$, we have $\mathbf{P}(|\|X\|^2 - d| \geq \varepsilon d) \leq 2e^{-d\varepsilon^2/8}$.

Proof. $\|X\|^2$ has a χ^2 distribution with expectation d, variance $2d$, and moment-generating function $\phi(t) = \mathbf{E}e^{t\|X\|^2} = (1-2t)^{-d/2}$.

(a) By Markov's inequality, for $t \in [0, \frac{1}{2}]$,

$$\mathbf{P}\left(\|X\|^2 \geq \lambda d\right) \leq \frac{\phi(t)}{e^{t\lambda d}};$$

the first assertion of the lemma follows by choosing $t = \frac{1}{2}(1 - \frac{1}{\lambda})$.

(b) Fix any $\varepsilon \in (0,1)$. Applying Markov's inequality tells us that for any $t \in [0, \frac{1}{2}]$,

$$\mathbf{P}(\|X\|^2 \geq (1+\varepsilon)d) = \mathbf{P}(e^{t\|X\|^2} \geq e^{t(1+\varepsilon)d}) \leq \frac{\phi(t)}{e^{t(1+\varepsilon)d}}$$

and for any $t \geq 0$,

$$\mathbf{P}(\|X\|^2 \leq (1-\varepsilon)d) = \mathbf{P}(e^{-t\|X\|^2} \geq e^{-t(1-\varepsilon)d}) \leq \phi(-t)e^{t(1-\varepsilon)d}.$$

Using $t = \frac{\varepsilon}{2(1+\varepsilon)}$ in the first case and $t = \frac{\varepsilon}{2(1-\varepsilon)}$ in the second yields bounds $e^{-d\varepsilon^2/4(1+\varepsilon)}$ and $e^{-d\varepsilon^2/4}$, respectively. ▮

Lemma 2 immediately implies that the distance between two points from the mixture is sharply concentrated around its conditional expected value. Here are the details.

Lemma 3 *If X and Y are chosen independently from $N(\mu_i, \sigma_i^2 I_d)$ and $N(\mu_j, \sigma_j^2 I_d)$ respectively, then for any $\varepsilon \in (0,1)$, the probability that $\|X-Y\|^2$ does not lie in the range*

$$\|\mu_i - \mu_j\|^2 + (\sigma_i^2 + \sigma_j^2)d(1 \pm \varepsilon) \pm 2\|\mu_i - \mu_j\|\sqrt{\sigma_i^2 + \sigma_j^2} \cdot \varepsilon d^{1/2}$$

is at most $2e^{-\varepsilon^2 d/8} + e^{-\varepsilon^2 d/2}$.

Proof. The sum of independent normals is itself normal. Specifically,

$$X - Y \overset{d}{=} N(\mu_i - \mu_j, (\sigma_i^2 + \sigma_j^2)I_d) \overset{d}{=} (\mu_i - \mu_j) + \sqrt{\sigma_i^2 + \sigma_j^2}\, W,$$

where W is a random variable with distribution $N(0, I_d)$. Therefore

$$\|X-Y\|^2 \overset{d}{=} \|\mu_i - \mu_j\|^2 + (\sigma_i^2 + \sigma_j^2)\|W\|^2 + 2\sqrt{\sigma_i^2 + \sigma_j^2}\,(\mu_i - \mu_j) \cdot W.$$

Lemma 2(b) gives a bound on $\|W\|^2$; for the last term we use

$$(\mu_i - \mu_j) \cdot W \overset{d}{=} \|\mu_i - \mu_j\|\, Z$$

where Z is standard normal and thus satisfies $\mathbf{P}(|Z| > \varepsilon d^{1/2}) \leq e^{-\varepsilon^2 d/2}$ (Durrett, p.7). ▮

Thus, for i.i.d. data from a high-dimensional mixture of spherical Gaussians, we have a very good idea of how interpoint distances will be distributed. Similarly, we can bound the number of points drawn from each Gaussian, and also the sizes of certain angles (dot products) formed by data points and cluster centers. The following lemma will be used repeatedly in the analysis.

Lemma 4 (Bounds on interpoint distances and angles, and cluster sizes) *Draw m data points from a c-separated mixture of k spherical Gaussians with smallest mixing weight at least w_{min}. Let $\sigma_{ij} = \max\{\sigma_i, \sigma_j\}$ and write the separation between Gaussians as $\|\mu_i - \mu_j\| = c_{ij}\sigma_{ij}\sqrt{d}$. Let S_i denote the points from the i^{th} Gaussian. Pick any $\varepsilon_o \in (0,1)$. Then, with probability at least $1 - (m^2+m)e^{-\varepsilon_o^2 d/8} - (\frac{1}{2}m^2 + km + km^2)e^{-\varepsilon_o^2 d/2} - ke^{-mw_{min}/8}$,*

(a) for any $x, y \in S_j$,

$$\|x-y\|^2 = 2\sigma_j^2 d(1 \pm \varepsilon_o);$$

(b) for $x \in S_i, y \in S_j, i \neq j$,

$$\|x-y\|^2 = (\sigma_i^2 + \sigma_j^2 + c_{ij}^2\sigma_{ij}^2)d(1 \pm 2\varepsilon_o).$$

(c) for any data point $y \in S_j$,

$$\|y-\mu_j\|^2 = \sigma_j^2 d(1 \pm \varepsilon_o)$$

while for $i \neq j$,

$$\|y-\mu_i\|^2 = (\sigma_j^2 + c_{ij}^2 \sigma_{ij}^2) d (1 \pm 2\varepsilon_o);$$

(d) For any $1 \leq i,j,g \leq k$ and any $x \in S_i, y \in S_g$,

$$|(x-\mu_i)\cdot(y-\mu_j)| \leq \sigma_i \varepsilon_o d \cdot \sqrt{(\sigma_g^2 + c_{jg}^2 \sigma_{jg}^2)(1+2\varepsilon_o)}.$$

(e) each $|S_i| \geq \frac{1}{2} m w_i$.

Proof. Part (a) follows immediately from the previous lemma. For (b), we start with

$$\|x-y\|^2 = (c_{ij}^2\sigma_{ij}^2 + \sigma_i^2 + \sigma_j^2)d \pm (\sigma_i^2 + \sigma_j^2 + 2c_{ij}\sigma_{ij}\sqrt{\sigma_i^2+\sigma_j^2})\varepsilon_o d$$

from the previous lemma and then simplify using $2c_{ij}\sigma_{ij}\sqrt{\sigma_i^2+\sigma_j^2} \leq c_{ij}^2\sigma_{ij}^2 + \sigma_i^2 + \sigma_j^2$. For the third claim, notice that $y-\mu_j \overset{d}{=} N(0, \sigma_j^2 I_d)$, so we can bound $\|y-\mu_j\|^2$ using Lemma 2. For the second half of (c),

$$\|y-\mu_i\|^2 = \|\mu_i - \mu_j\|^2 + \|y-\mu_j\|^2 - 2(\mu_i - \mu_j)\cdot(y-\mu_j).$$

As was done in the proof of Lemma 3, we can show that $(\mu_i-\mu_j)\cdot(y-\mu_j)$ has the same distribution as $\|\mu_i-\mu_j\|$ times a $N(0,\sigma_j^2)$ random variable. Putting the pieces together, $\|y-\mu_i\|^2 = (c_{ij}^2\sigma_{ij}^2 + \sigma_j^2)d \pm (\sigma_j^2 + 2c_{ij}\sigma_{ij}\sigma_j)\varepsilon_o d$, finishing up with the inequality $2c_{ij}\sigma_{ij}\sigma_j \leq c_{ij}^2\sigma_{ij}^2 + \sigma_j^2$.

For part (d), imagine that first y is chosen, then x. Since $x-\mu_i \overset{d}{=} N(0,\sigma_i^2 I_d)$, the component of $x-\mu_i$ in the direction of $y-\mu_j$ (after y is fixed) has distribution $N(0,\sigma_i^2)$, and thus has absolute value $\leq \sigma_i\varepsilon_o d^{1/2}$ with probability at least $1-e^{-\varepsilon_o^2 d/2}$. Whereupon

$$|(x-\mu_i)\cdot(y-\mu_j)| \;\leq\; \|y-\mu_j\|\sigma_i\varepsilon_o d^{1/2} \;\leq\; \sqrt{(\sigma_g^2 + c_{jg}^2\sigma_{jg}^2)d(1+2\varepsilon_o)}\cdot\sigma_i\varepsilon_o d^{1/2}.$$

Finally, (e) follows from the Chernoff bound; see, for instance, the appendix of Kearns and Vazirani (1991):

$$\mathbf{P}(|S_i| \leq \tfrac{1}{2} m w_i) \;\leq\; e^{-mw_i/8}.$$

∎

Next, we turn to concentration properties of means of subsets of points. We'll start by showing that there is no large subset of S_i whose average is far from μ_i.

Lemma 5 (Averages of subsets) *Pick a set of n points randomly from $N(0,I_d)$. Choose any integer $1 \leq t \leq n$. Then with probability at least $1-e^{-d/2}$, no subset of t or more of the points has mean of norm greater than $\varepsilon\sqrt{d}$, where*

$$\varepsilon = \sqrt{6\max\left(\frac{1}{t}, \frac{1}{d}\ln\frac{ne}{t}\right)}.$$

Proof. First observe that it suffices to prove the statement for subsets of size exactly t.

Fix any set of t indices. The mean of the corresponding points, call it μ, is distributed according to $N(0, \frac{1}{t}I_d) \overset{d}{=} t^{-1/2}N(0, I_d)$. In particular, $\mathbf{E}\|\mu\|^2 = d/t$. Lemma 2 tells us that for any $\lambda > 1$,

$$\mathbf{P}(\|\mu\|^2 > \lambda \cdot d/t) \leq \exp(-d(\lambda - 1 - \ln\lambda)/2).$$

We will choose $\lambda = t\varepsilon^2$, where ε is defined in the lemma statement. This guarantees that $\lambda \geq 6$ and thus $1 + \ln\lambda \leq \frac{1}{2}\lambda$, whereupon

$$\mathbf{P}(\|\mu\|^2 > \varepsilon^2 d) \ \leq \ e^{-d\lambda/4} \ = \ e^{-dt\varepsilon^2/4}.$$

The number of possible choices of t indices is $\binom{n}{t} \leq (ne/t)^t$. Summing over these,

$$\mathbf{P}(\exists \text{ subset of } t \text{ points with } \|\text{mean}\|^2 > \varepsilon^2 d) \ \leq \ \left(\frac{ne}{t}\right)^t e^{-dt\varepsilon^2/4} \ \leq \ e^{-dt\varepsilon^2/12} \ \leq \ e^{-d/2},$$

by the particular choice of ε. ▮

This will enable us to show that if one of EM's cluster-estimates overlaps substantially with a true cluster S_i, then the mean of the overlapping points will be close to μ_i. The only technical difficulty is that EM has soft assignments for data points, and therefore we need to also deal with *weighted* averages.

More generally, we consider the following problem: suppose you are allowed to distribute a specific amount of weight over the elements of S_i, where each element receives a weight between 0 and 1. What is the *worst* soft assignment of this kind, the one whose weighted average is furthest from μ_i? It is not difficult to see that the worst assignment is a *hard* assignment, whereupon we can apply the previous lemma.

Lemma 6 (Weighted averages) *For any finite set of points $S \subset \mathbb{R}^d$, with associated $[0,1]$-valued weights $\{w_x : x \in S\}$, there is a subset $T \subset S$ such that*

1. $|T| = \lfloor \sum_{x \in S} w_x \rfloor$*; and*
2. $\|mean(T)\| \geq \|weighted\text{-}mean(S)\|$*, that is,*

$$\left\| \frac{1}{|T|} \sum_{x \in T} x \right\| \ \geq \ \left\| \frac{\sum_{x \in S} w_x x}{\sum_{x \in S} w_x} \right\|.$$

Proof. Let μ_S denote the weighted mean of S. Order points $x \in S$ by increasing $x \cdot \mu_S$, and let T consist of the last $\lfloor \sum_{x \in S} w_x \rfloor$ points in this ordering. Then, the component of mean(T) in the direction of μ_S is least as large as that of any weighted mean of the points in which the weights are in the range $[0,1]$ and sum to $\geq \lfloor \sum_{x \in S} w_x \rfloor$. Letting $\mu_T = \text{mean}(T)$, in particular $\mu_T \cdot \mu_S \geq \mu_S \cdot \mu_S = \|\mu_S\|^2$; and thus $\|\mu_T\|^2$ is at least this large. ▮

Remark In what follows we will assume that all the large deviation bounds of this section—Lemmas 4, 5, and 6—hold for the particular sample S we have drawn, for some $\varepsilon_o \in (0,1)$. For Lemma 5, we will use $t = \frac{1}{2}mw_T$.

From these various concentration properties, we see that points from the same Gaussian, say the i^{th} one, are at distance about $\sqrt{2\sigma_i^2 d}$ from each other while points from different Gaussians $i \neq j$ are at distance about $\sqrt{(\sigma_i^2 + \sigma_j^2 + c_{ij}^2\sigma_{ij}^2)d}$ from each other. These estimates are accurate to within $O(\sigma_{ij}d^{1/4})$. Therefore, in the case where all Gaussians have the same variance, it is sufficient to have $c_{ij}^2\sigma_{ij}^2 d \gg \sigma_{ij}^2 d^{1/2}$—more simply, $c \gg d^{-1/4}$—for the interpoint distances to reveal enough information for clustering. In particular, it should be possible to make EM work well. The general case, in which the Gaussians have different variances, requires more careful treatment but yields the same conclusion.

4. Conditions

Various parts of the analysis require assumptions on the sample size, dimensionality, and separation. To simplify the exposition, we summarize these conditions up front.

(C1) $\|\mu_i - \mu_j\|^2 \geq |\sigma_i^2 - \sigma_j^2| d$ for all i, j .

(C2) $\varepsilon_o \leq \frac{1}{96}\min(1, c^2)$.

(C3) $d \geq 864 \max(1, c^{-2}) \ln 8el$.

(C4) $m \geq 6912 l \max(1, c^{-2})$.

The constants are astronomical but are doubtless much larger than they need to be, as no attempt has been made to optimize them. For (C2), recall that ε_o is the extent to which squared interpoint distances are concentrated: by Lemma 4, these are all within a multiplicative factor $1 \pm O(\varepsilon_o)$ of their expected values.

We start by establishing properties of the initial choice of centers and variances.

5. Initialization

As we saw earlier, it is crucial that every cluster is represented in the initial center-estimates, and that the variance-estimates are fairly accurate. We now confirm these conditions.

Lemma 7 (Properties of the initial parameters) *If $l > k$ and each $w_i \geq w_{min}$ and condition (C1) holds, then with probability at least $1 - k(l+1)e^{-lw_{min}} - ke^{-lw_{min}/12}$,*
(a) every Gaussian is represented at least twice in the initial center-estimates;
(b) the i^{th} Gaussian provides at most $\frac{3}{2}lw_i$ initial center-estimates, for all $1 \leq i \leq k$; and
(c) if the r^{th} center-estimate is drawn from S_i, then $\sigma_i^2(1-2\varepsilon_o) \leq \sigma_r^{\langle 0 \rangle 2} \leq \sigma_i^2(1+\varepsilon_o)$.

Proof. Assume the center-estimates are simply the first l (randomly chosen) data points. The chance that these do not touch a particular S_i at least twice is $(1-w_i)^l + lw_i(1-w_i)^{l-1} \leq (l+1)(1-w_{min})^l \leq (l+1)e^{-lw_{min}}$. Similarly, since the number of center-estimates chosen from S_i has expectation lw_i, a Chernoff bound tells us that

$$\mathbf{P}(\text{there are more than } \tfrac{3}{2}lw_i \text{ initial centers from } S_i) \quad \leq \quad e^{-lw_i/12}.$$

For the bound on $\sigma_r^{\langle 0\rangle}$, we have already established that there is at least one other center-estimate from the same S_i. If this is the closest center-estimate to $\mu_r^{\langle 0\rangle}$, then by Lemma 4(a) the squared distance between the two is $2\sigma_i^2 d(1\pm\varepsilon_o)$. On the other hand, the closest center-estimate to $\mu_r^{\langle 0\rangle}$ might be from some other cluster S_j, in which case, by Lemma 4(b), the squared distance is at least $(\sigma_i^2+\sigma_j^2+c_{ij}^2\sigma_{ij}^2)d(1-2\varepsilon_o) \geq 2\sigma_i^2 d(1-2\varepsilon_o)$, since by (C1), $c_{ij}^2\sigma_{ij}^2 \geq |\sigma_i^2-\sigma_j^2|$. These two cases give the upper and lower bounds on $\sigma_r^{\langle 0\rangle 2}$. ∎

Remark All the theorems of the following sections are made under the additional hypothesis that the high-probability events of Lemma 7 hold.

6. The First Round of EM

What happens during the first round of EM? The first thing we clarify is that although in principle EM allows "soft" assignments in which each data point is fractionally distributed over various clusters, in practice for large d every data point will give almost its entire weight to center-estimates from one cluster. This is because in high dimension, the distances between clusters are so great that there is just a very narrow region between two clusters where there is any ambiguity of assignment, and the probability that points fall within this region is miniscule.

Lemma 8 (Soft assignments) *Suppose $\mu_{i'}^{\langle 0\rangle} \in S_i$ and $\mu_{j'}^{\langle 0\rangle} \in S_j$. If condition (C2) holds, then for any data point $x \in S_i$, the ratio between the probabilities assigned to x by Gaussian-estimates $p_{i'}^{\langle 1\rangle}$ and $p_{j'}^{\langle 1\rangle}$ is*

$$\frac{p_{i'}^{\langle 1\rangle}(x)}{p_{j'}^{\langle 1\rangle}(x)} \geq \exp\left(\frac{c_{ij}^2\sigma_{ij}^2}{\sigma_j^2}\cdot\frac{d}{8}\right).$$

Proof. The following calculations make occasional use of Lemma 4, along with several inequalities that exploit the bound (C2) on ε_o.

$$\begin{aligned}
\frac{p_{i'}^{\langle 1\rangle}(x)}{p_{j'}^{\langle 1\rangle}(x)} &= \frac{\sigma_{j'}^{\langle 0\rangle d}}{\sigma_{i'}^{\langle 0\rangle d}}\exp\left\{\frac{\|x-\mu_{j'}^{\langle 0\rangle}\|^2}{2\sigma_{j'}^{\langle 0\rangle 2}} - \frac{\|x-\mu_{i'}^{\langle 0\rangle}\|^2}{2\sigma_{i'}^{\langle 0\rangle 2}}\right\} \\
&\geq \exp\left\{\frac{d}{2}\ln\frac{\sigma_j^2(1-2\varepsilon_o)}{\sigma_i^2(1+\varepsilon_o)} + \frac{(\sigma_i^2+\sigma_j^2+c_{ij}^2\sigma_{ij}^2)d\cdot(1-2\varepsilon_o)}{2\sigma_j^2(1+\varepsilon_o)} - \frac{2\sigma_i^2 d(1+\varepsilon_o)}{2\sigma_i^2(1-2\varepsilon_o)}\right\} \\
&\geq \exp\left\{d\left(\frac{1}{2}\ln\frac{\sigma_j^2}{\sigma_i^2}(1-3\varepsilon_o) + \frac{\sigma_i^2+\sigma_j^2+c_{ij}^2\sigma_{ij}^2}{2\sigma_j^2}(1-3\varepsilon_o) - 1 - \frac{7\varepsilon_o}{2}\right)\right\} \\
&\geq \exp\left\{d\left(\ln(1-3\varepsilon_o) + \frac{c_{ij}^2\sigma_{ij}^2}{2\sigma_j^2}(1-3\varepsilon_o) - 5\varepsilon_o\right)\right\} \quad \geq \exp\left(\frac{c_{ij}^2\sigma_{ij}^2}{\sigma_j^2}\cdot\frac{d}{8}\right).
\end{aligned}$$

The second-last step uses $\ln(\sigma_j^2/\sigma_i^2) + (\sigma_i^2/\sigma_j^2)(1-3\varepsilon_o) \geq 1+\ln(1-3\varepsilon_o)$, which can be obtained easily from the more familiar $x \geq 1+\ln x$. ∎

In the case where all the Gaussians have the same variance, this lemma says that each data point is given weight at most $e^{-c^2 d/8}$ by any center-estimate from a different cluster. When the Gaussians do not have the same variance, the lemma is even stronger. In particular, if there is a small cluster S_i right near a large one, S_j, we can conclude that center-estimates from the small cluster assign weight at most $e^{-c^2\sigma_j^2 d/\sigma_i^2}$ to points from the big cluster. We need this stronger bound in our subsequent analysis: the two clusters could be at such different scales that a point from the large cluster, because it has substantial norm, could significantly throw off the center-estimate from the small cluster.

We are now in a position to assess what happens during the first round of EM. At the end of this round, let C_j denote the center-estimates originally from S_j which have high mixing weight, that is, $C_j = \{\mu_{j'}^{\langle 1\rangle} : \mu_{j'}^{\langle 0\rangle} \in S_j, w_{j'}^{\langle 1\rangle} \geq w_T\}$. We will see that such center-estimates move quite a bit closer to their respective true centers μ_j as a result of the first EM update.

Lemma 9 *Under conditions (C2), (C3), and (C4), any "non-starved" center-estimate $\mu_{i'}^{\langle 1\rangle} \in C_i$ has*

$$\|\mu_{i'}^{\langle 1\rangle} - \mu_i\| \leq \frac{1}{8}\min(1,c)\sigma_i\sqrt{d}.$$

Proof. Bound $\|\mu_{i'}^{\langle 1\rangle} - \mu_i\|$ by the sum of two terms:

$$\begin{aligned}
\|\mu_{i'}^{\langle 1\rangle} - \mu_i\| &= \left\| \frac{\sum_{x\in S} p_{i'}^{\langle 1\rangle}(x)(x-\mu_i)}{\sum_{x\in S} p_{i'}^{\langle 1\rangle}(x)} \right\| \\
&\leq \frac{\|\sum_{x\in S_i} p_{i'}^{\langle 1\rangle}(x)(x-\mu_i)\| + \|\sum_{x\notin S_i} p_{i'}^{\langle 1\rangle}(x)(x-\mu_i)\|}{\sum_{x\in S} p_{i'}^{\langle 1\rangle}(x)} \\
&\leq \frac{\|\sum_{x\in S_i} p_{i'}^{\langle 1\rangle}(x)(x-\mu_i)\|}{\sum_{x\in S_i} p_{i'}^{\langle 1\rangle}(x)} + \frac{\|\sum_{j\neq i}\sum_{x\in S_j} p_{i'}^{\langle 1\rangle}(x)(x-\mu_i)\|}{\sum_{x\in S} p_{i'}^{\langle 1\rangle}(x)}.
\end{aligned}$$

The first term can be bounded using Lemma 5, provided $p_{i'}^{\langle 1\rangle}(S_i)$ is substantial. By Lemma 8, for any $x \in S_j, j \neq i$, we have $p_{i'}^{\langle 1\rangle}(x) \leq e^{-c_{ij}^2\sigma_{ij}^2 d/8\sigma_i^2} \leq e^{-c^2 d/8}$. Thus

$$\sum_{x\in S_i} p_{i'}^{\langle 1\rangle}(x) \;\geq\; \sum_{x\in S} p_{i'}^{\langle 1\rangle}(x) - \sum_{j\neq i}\sum_{x\in S_j} p_{i'}^{\langle 1\rangle}(x) \;\geq\; mw_T - me^{-c^2 d/8} \;\geq\; \tfrac{1}{2}mw_T + 1$$

using (C3) and (C4). Lemmas 5 and 6, together with (C4), then give

$$\frac{\|\sum_{x\in S_i} p_{i'}^{\langle 1\rangle}(x)(x-\mu_i)\|}{\sum_{x\in S_i} p_{i'}^{\langle 1\rangle}(x)} \;\leq\; \sigma_i\sqrt{d}\cdot\sqrt{6\max\left(\frac{2}{mw_T}, \frac{1}{d}\ln\frac{2e|S_i|}{mw_T}\right)} \;\leq\; \frac{1}{12}\min(1,c)\sigma_i\sqrt{d}.$$

For the second half of the $\|\mu_{i'}^{\langle 1\rangle} - \mu_i\|$ expression, we observe that for any $x \in S_j, j \neq i$, by Lemma 4 $\|x-\mu_i\| \leq \sqrt{(\sigma_j^2 + c_{ij}^2\sigma_{ij}^2)d(1+2\varepsilon_o)}$. Using the conditions on d and ε_o, this can be upper-bounded by $e^{c_{ij}^2\sigma_{ij}^2 d/16\sigma_i^2}\sigma_i\sqrt{d}$. Thus $p_{i'}^{\langle 1\rangle}(x)\|x-\mu_i\| \leq e^{-c^2 d/16}\sigma_i\sqrt{d}$, and

$$\begin{aligned}
\frac{\|\sum_{j\neq i}\sum_{x\in S_j} p_{i'}^{\langle 1\rangle}(x)(x-\mu_i)\|}{\sum_{x\in S} p_{i'}^{\langle 1\rangle}(x)} &\leq \frac{1}{mw_T}\sum_{j\neq i}\sum_{x\in S_j} p_{i'}^{\langle 1\rangle}(x)\|x-\mu_i\| \\
&\leq \frac{1}{w_T}e^{-c^2 d/16}\sigma_i\sqrt{d} \;\leq\; \frac{1}{24}\min(1,c)\sigma_i\sqrt{d},
\end{aligned}$$

completing the proof. ▮

We also need to analyze the variance-estimates. These started off excellent, and so we mostly need to check that they don't degrade too much during the first round of EM. The difficulty with the usual formula for variance is that it involves terms of the form $\|x-\mu_i^{\langle 1\rangle}\|^2$, whereas we only have tight concentration bounds for terms like $\|x-\mu_i\|^2$. To cope with this, we first derive an alternative expression for the variance.

Lemma 10 (Alternative formula for variance) *For any i, and any choice of $\mu \in \mathbb{R}^d$, the formula for $\sigma_i^{\langle t\rangle 2}$ can be rewritten thus:*

$$\sigma_i^{\langle t\rangle 2} \;=\; \frac{\sum_x p_i^{\langle t\rangle}(x)\|x-\mu_i^{\langle t\rangle}\|^2}{d\sum_x p_i^{\langle t\rangle}(x)} \;=\; \frac{\sum_x p_i^{\langle t\rangle}(x)\|x-\mu\|^2}{d\sum_x p_i^{\langle t\rangle}(x)} - \frac{\|\mu-\mu_i^{\langle t\rangle}\|^2}{d}.$$

Proof. Consider the distribution over S which assigns point $x\in S$ a probability mass proportional to $p_i^{\langle t\rangle}(x)$. Taking expectations over X drawn from this distribution, we have $\mathbb{E}X = \mu_i^{\langle t\rangle}$, and for any $\mu\in\mathbb{R}^d$,

$$\mathbb{E}\|X-\mu\|^2 \;=\; \mathbb{E}\|X\|^2 + \|\mu\|^2 - 2\mu\cdot\mathbb{E}X \;=\; \mathbb{E}\|X\|^2 + \|\mu\|^2 - 2\mu\cdot\mu_i^{\langle t\rangle}$$

and similarly

$$\mathbb{E}\|X-\mu_i^{\langle t\rangle}\|^2 \;=\; \mathbb{E}\|X\|^2 + \|\mu_i^{\langle t\rangle}\|^2 - 2\mu_i^{\langle t\rangle}\cdot\mathbb{E}X \;=\; \mathbb{E}\|X\|^2 - \|\mu_i^{\langle t\rangle}\|^2.$$

Subtracting,

$$\mathbb{E}\|X-\mu\|^2 - \mathbb{E}\|X-\mu_i^{\langle t\rangle}\|^2 \;=\; \|\mu\|^2 - 2\mu\cdot\mu_i^{\langle t\rangle} + \|\mu_i^{\langle t\rangle}\|^2 \;=\; \|\mu-\mu_i^{\langle t\rangle}\|^2,$$

which is a paraphrase of the lemma statement. ▮

It is now simpler to bound the variances at the end of first round of EM.

Lemma 11 (Variance estimates in round one) *Under conditions (C2), (C3), and (C4), for any $i'\in C_i$,*

$$\sigma_i^2\left(1-\frac{3}{64}\min(1,c^2)\right) \;\le\; \sigma_{i'}^{\langle 1\rangle 2} \;\le\; \sigma_i^2\left(1+\frac{1}{32}\right).$$

Proof. By Lemma 10,

$$\sigma_{i'}^{\langle 1\rangle 2} \;=\; \frac{\sum_{x\in S} p_{i'}^{\langle 1\rangle}(x)\|x-\mu_i\|^2}{d\sum_{x\in S} p_{i'}^{\langle 1\rangle}(x)} - \frac{\|\mu_i-\mu_{i'}^{\langle 1\rangle}\|^2}{d}.$$

First let us lower-bound this, using Lemmas 4(c), 8, and 9.

$$\begin{aligned}
\sigma_{i'}^{\langle 1\rangle 2} &\geq \frac{\sum_{x\in S_i} p_{i'}^{\langle 1\rangle}(x)\|x-\mu_i\|^2}{d\sum_{x\in S} p_{i'}^{\langle 1\rangle}(x)} - \frac{\|\mu_i-\mu_{i'}^{\langle 1\rangle}\|^2}{d} \\
&\geq \frac{\sum_{x\in S_i} p_{i'}^{\langle 1\rangle}(x)\cdot\sigma_i^2 d(1-\varepsilon_o)}{d\sum_{x\in S} p_{i'}^{\langle 1\rangle}(x)} - \frac{\min(1,c^2)}{64}\sigma_i^2 \\
&\geq \sigma_i^2(1-\varepsilon_o)\cdot\frac{p_{i'}^{\langle 1\rangle}(S)-\sum_{j\neq i} p_{i'}^{\langle 1\rangle}(S_j)}{p_{i'}^{\langle 1\rangle}(S)} - \frac{\min(1,c^2)}{64}\sigma_i^2 \\
&\geq \sigma_i^2(1-\varepsilon_o)\cdot\left(1-\frac{me^{-c^2d/8}}{mw_T}\right) - \frac{\min(1,c^2)}{64}\sigma_i^2.
\end{aligned}$$

For the upper bound, we again use Lemmas 4(c) and 8, along with the conditions on d and ε_o, to assert that for points $x\in S_j, j\neq i$, we have $p_{i'}^{\langle 1\rangle}(x)\|x-\mu_i\|^2 \leq e^{-c_{ij}^2\sigma_{ij}^2 d/8\sigma_i^2}(\sigma_j^2+c_{ij}^2\sigma_{ij}^2)d\cdot(1+2\varepsilon_o) \leq e^{-c^2d/16}\sigma_i^2 d$, and thus

$$\begin{aligned}
\sigma_{i'}^{\langle 1\rangle 2} &\leq \frac{\sum_{x\in S_i} p_{i'}^{\langle 1\rangle}(x)\|x-\mu_i\|^2 + \sum_{j\neq i}\sum_{x\in S_j} p_{i'}^{\langle 1\rangle}(x)\|x-\mu_i\|^2}{d\sum_{x\in S} p_{i'}^{\langle 1\rangle}(x)} \\
&\leq \frac{\sum_{x\in S_i} p_{i'}^{\langle 1\rangle}(x)\cdot\sigma_i^2 d(1+\varepsilon_o)}{dp_{i'}^{\langle 1\rangle}(S)} + \frac{\sum_{j\neq i}\sum_{x\in S_j} e^{-c^2d/16}\sigma_i^2 d}{dp_{i'}^{\langle 1\rangle}(S)} \\
&\leq \sigma_i^2(1+\varepsilon_o) + \frac{me^{-c^2d/16}\sigma_i^2}{mw_T}.
\end{aligned}$$

The rest follows by substituting in conditions (C2) and (C3). ▌

Remark Henceforth we will assume that the conclusions of Lemmas 9 and 11 hold.

7. Pruning

We now know each center-estimate in C_j is accurate within $\frac{1}{8}\min(1,c)\sigma_j\sqrt{d}$. A simple clustering heuristic due to Gonzalez (1985), described in Section 2.4, is used to choose k points from $\cup_j C_j$.

Lemma 12 *Under condition (C3), the sets C_i obey the following properties.*
(a) Each C_i is non-empty.
(b) There is a real value $\Delta > 0$ such that if $x\in C_i$ and $y,z\in C_j$ $(i\neq j)$ then $d(y,z)\leq\Delta$ and $d(x,y)>\Delta$.
(c) The pruning procedure identifies exactly one member of each C_i.

Proof. (a) From Lemmas 4 and 7 we know that $|S_i|\geq\frac{1}{2}mw_i$, and that at most $\frac{3}{2}lw_i$ initial center-estimates are chosen from S_i. It was seen in Lemma 9 that each point in S_i gives weight at least $1-le^{-c^2d/8}$ to center-estimates from S_i. It follows that at the end of the first round of EM, at least one of these center-estimates must have mixing weight at least

$$\frac{(\frac{1}{2}mw_i)(1-le^{-c^2d/8})}{m\cdot\frac{3}{2}lw_i} = \frac{1}{3l}\cdot(1-le^{-c^2d/8}) \geq \frac{1}{4l} = w_T$$

and therefore C_i cannot be empty.
(b) Pick $\mu_{i'}^{\langle 1\rangle} \in C_i$ and $\mu_{j'}^{\langle 1\rangle}, \mu_{j''}^{\langle 1\rangle} \in C_j$ for any pair $i \neq j$. Then, from Lemmas 9 and 11,

$$d(\mu_{j'}^{\langle 1\rangle}, \mu_{j''}^{\langle 1\rangle}) = \frac{\|\mu_{j'}^{\langle 1\rangle} - \mu_{j''}^{\langle 1\rangle}\|}{\sigma_{j'}^{\langle 1\rangle} + \sigma_{j''}^{\langle 1\rangle}} < \frac{2 \cdot \frac{c}{8}\sigma_j\sqrt{d}}{2 \cdot \frac{61}{64}\sigma_j} < \frac{c}{6}\sqrt{d}.$$

Call this value Δ. Meanwhile,

$$\begin{aligned} d(\mu_{i'}^{\langle 1\rangle}, \mu_{j'}^{\langle 1\rangle}) = \frac{\|\mu_{i'}^{\langle 1\rangle} - \mu_{j'}^{\langle 1\rangle}\|}{\sigma_{i'}^{\langle 1\rangle} + \sigma_{j'}^{\langle 1\rangle}} &> \frac{c_{ij}\sigma_{ij}\sqrt{d} - \frac{c}{8}\sigma_i\sqrt{d} - \frac{c}{8}\sigma_j\sqrt{d}}{\frac{33}{32}\sigma_i + \frac{33}{32}\sigma_j} \\ &\geq \frac{c_{ij} \cdot \frac{1}{2}(\sigma_i + \sigma_j)\sqrt{d} - \frac{c}{8}(\sigma_i + \sigma_j)\sqrt{d}}{\frac{33}{32}(\sigma_i + \sigma_j)} \geq \frac{c}{3}\sqrt{d}, \end{aligned}$$

strictly greater than Δ.
(c) There are k true clusters and the pruning procedure picks exactly k center-estimates. It will not pick two from the same true cluster because these must be at distance $\leq \Delta$ from each other, whereas there must be some untouched cluster containing a center-estimate at distance $> \Delta$ from all points selected thus far. ∎

8. The Second Round of EM

We now have one center-estimate $\mu_i^{\langle 1\rangle}$ per true cluster (for convenience permute their labels to match the S_i), each with mixing weight $\frac{1}{k}$ and covariance $\sigma_i^{\langle 1\rangle 2} I_d$. Furthermore each $\mu_i^{\langle 1\rangle}$ is within distance $\frac{1}{8}\min(1, c)\sigma_i\sqrt{d}$ of the corresponding true Gaussian center μ_i. Such favorable circumstances make it easy to show that the subsequent round of EM achieves near-perfect clustering.

There is just one tricky issue. As in the first round, in order to bound EM's soft assignments, we'd like to assert that the distances $\|x - \mu_i^{\langle 1\rangle}\|^2$ are tightly concentrated around certain values. What is different this time, however, is the statistical dependency between data points x and center-estimates $\mu_i^{\langle 1\rangle}$. To avoid having to manage this dependency, we instead recall that the $\mu_i^{\langle 1\rangle}$ are just weighted averages of data points, and as far as possible, we rewrite expressions like $\|x - \mu_i^{\langle 1\rangle}\|^2$ as weighted sums of expressions involving only data points, such as $\|x - y\|^2$ or $x \cdot y$. There is one particular kind of dot product which will be an especially useful building block, and we start by analyzing it.

Lemma 13 *Pick any $1 \leq i, j \leq k$. Under conditions (C2) and (C3), for any $x \in S_i$,*

$$|(x - \mu_i) \cdot (\mu_j^{\langle 1\rangle} - \mu_j)| \leq \frac{3}{2}\varepsilon_o \sigma_{ij}^2 d.$$

Proof. We could use the fact that $\mu_j^{\langle 1\rangle}$ is reasonably close to μ_j, but this doesn't give a very good bound. Instead we notice that $(x - \mu_i) \cdot (\mu_j^{\langle 1\rangle} - \mu_j)$ is a weighted average of terms of the form $(x - \mu_i) \cdot (y - \mu_j)$, as y varies over S. When $y \in S_j$, these dot products are small, as seen in Lemma 4(d). And when $y \notin S_j$, the weight of the term is small.

Specifically, suppose $y \in S_g$ for $g \neq j$. By Lemmas 4(d) and 8,

$$p_j^{\langle 1\rangle}(y)\,|(x-\mu_i)\cdot(y-\mu_j)| \leq e^{-c_{jg}^2\sigma_{jg}^2 d/8\sigma_j^2}\cdot\sqrt{(\sigma_g^2+c_{jg}^2\sigma_{jg}^2)(1+2\varepsilon_o)}\cdot\sigma_i\varepsilon_o d.$$

By the conditions on d, this is at most $e^{-c^2d/16}\sigma_i\sigma_j\varepsilon_o d$. Hence

$$\begin{aligned}
|(x-\mu_i)\cdot(\mu_j^{\langle 1\rangle}-\mu_j)| &= \left|\frac{\sum_y p_j^{\langle 1\rangle}(y)(x-\mu_i)\cdot(y-\mu_j)}{p_j^{\langle 1\rangle}(S)}\right| \\
&\leq \frac{\sum_{y\in S_j} p_j^{\langle 1\rangle}(y)|(x-\mu_i)\cdot(y-\mu_j)|}{p_j^{\langle 1\rangle}(S_j)} + \frac{\sum_{g\neq j}\sum_{y\in S_g} p_j^{\langle 1\rangle}(y)|(x-\mu_i)\cdot(y-\mu_j)|}{p_j^{\langle 1\rangle}(S)} \\
&\leq \sigma_i\sigma_j\varepsilon_o d\sqrt{1+2\varepsilon_o} + \frac{1}{mw_T}\sum_{g\neq j}\sum_{y\in S_g} e^{-c^2d/16}\sigma_i\sigma_j\varepsilon_o d \\
&\leq \varepsilon_o\sigma_{ij}^2 d\cdot\left(1+\varepsilon_o+\frac{1}{w_T}e^{-c^2d/16}\right).
\end{aligned}$$

Under (C2) and (C3), the term in parentheses is at most $3/2$. ∎

We now develop a counterpart of Lemma 8, a bound on how "soft" EM's assignments can be.

Lemma 14 *Under conditions (C1), (C2), and (C3), for any $1 \leq i \neq j \leq k$, and for any $x \in S_i$,*

$$\frac{p_i^{\langle 2\rangle}(x)}{p_j^{\langle 2\rangle}(x)} \geq \exp\left(\frac{c_{ij}^2\sigma_{ij}^2 d}{8\sigma_j^2}\right).$$

Proof. This is mostly a matter of confirming that $\|x-\mu_i^{\langle 1\rangle}\|, \|x-\mu_j^{\langle 1\rangle}\|, \sigma_i^{\langle 1\rangle}, \sigma_j^{\langle 1\rangle}$ are not too far from $\|x-\mu_i\|, \|x-\mu_j\|, \sigma_i, \sigma_j$. Making use of Lemmas 4(c), 9, and 13,

$$\begin{aligned}
\|x-\mu_i^{\langle 1\rangle}\|^2 &= \|x-\mu_i\|^2 + \|\mu_i-\mu_i^{\langle 1\rangle}\|^2 + 2(x-\mu_i)\cdot(\mu_i-\mu_i^{\langle 1\rangle}) \\
&\leq \sigma_i^2 d(1+\varepsilon_o) + \frac{\min(1,c^2)}{64}\sigma_i^2 d + 3\varepsilon_o\sigma_i^2 d \\
&\leq \sigma_i^2 d\left(1+\frac{1}{16}\min(1,c^2)\right)
\end{aligned}$$

In much the same vein,

$$\begin{aligned}
\|x-\mu_j^{\langle 1\rangle}\|^2 &= \|x-\mu_j\|^2 + \|\mu_j-\mu_j^{\langle 1\rangle}\|^2 + 2((x-\mu_i)+(\mu_i-\mu_j))\cdot(\mu_j-\mu_j^{\langle 1\rangle}) \\
&\geq (\sigma_i^2+c_{ij}^2\sigma_{ij}^2)d(1-2\varepsilon_o) - 3\varepsilon_o\sigma_{ij}^2 d - 2c_{ij}\sigma_{ij}d^{1/2}\cdot\frac{\min(1,c)}{8}\sigma_j d^{1/2} \\
&\geq \sigma_i^2 d\left(1-\frac{1}{48}\min(1,c^2)\right) + c_{ij}^2\sigma_{ij}^2 d\left(1-\frac{29}{96}\right).
\end{aligned}$$

Using these and the variance bound from Lemma 11, the rest follows in the same manner as Lemma 8. ∎

Henceforth, assume this condition holds true. The best we can now hope for is that each final estimate $\mu_i^{\langle 2\rangle}$ is exactly the mean of S_i. In fact, it will turn out that the error $\|\mu_i^{\langle 2\rangle}-\mu_i\|$ is not too different from $\|\text{mean}(S_i)-\mu_i\|$.

Lemma 15 *Under conditions (C1)–(C4), for each i,*

$$\|\mu_i^{\langle 2\rangle} - \mu_i\| \;\le\; \|\text{mean}(S_i) - \mu_i\| + \frac{5}{w_{min}} e^{-c^2 d/16} \sigma_i \sqrt{d}.$$

Proof. As usual, we start by separating points in S_i from those outside.

$$\begin{aligned}
\|\mu_i^{\langle 2\rangle} - \mu_i\| &= \left\| \frac{\sum_{x\in S} p_i^{\langle 2\rangle}(x)(x-\mu_i)}{\sum_{x\in S} p_i^{\langle 2\rangle}(x)} \right\| \\
&\le \frac{1}{p_i^{\langle 2\rangle}(S)} \left\| \sum_{x\in S_i} p_i^{\langle 2\rangle}(x)(x-\mu_i) \right\| + \frac{1}{p_i^{\langle 2\rangle}(S)} \sum_{j\ne i} \sum_{x\in S_j} p_i^{\langle 2\rangle}(x)\|x-\mu_i\|.
\end{aligned}$$

With Lemma 14 in hand, these two terms are straightforward to bound. For instance, we know that for any $x \in S_i$, $p_i^{\langle 2\rangle}(x) \ge 1 - ke^{-c^2 d/8}$. Hence the first term is at most

$$\begin{aligned}
&\frac{1}{p_i^{\langle 2\rangle}(S)} \left(\left\| \sum_{x\in S_i} (1-ke^{-c^2d/8})(x-\mu_i) \right\| + \left\| \sum_{x\in S_i} (p_i^{\langle 2\rangle}(x) - (1-ke^{-c^2d/8}))(x-\mu_i) \right\| \right) \\
&\le \frac{\left\| \sum_{x\in S_i} (1-ke^{-c^2d/8})(x-\mu_i) \right\|}{|S_i| \cdot (1-ke^{-c^2d/8})} + \frac{1}{p_i^{\langle 2\rangle}(S)} \sum_{x\in S_i} ke^{-c^2d/8} \|x-\mu_i\| \\
&\le \|\text{mean}(S_i) - \mu_i\| + \frac{|S_i| \cdot ke^{-c^2d/8}}{|S_i| \cdot (1-ke^{-c^2d/8})} \sigma_i \sqrt{d(1+2\varepsilon_o)} \\
&\le \|\text{mean}(S_i) - \mu_i\| + 2ke^{-c^2d/8} \sigma_i \sqrt{d}.
\end{aligned}$$

while the second term is

$$\begin{aligned}
\frac{1}{p_i^{\langle 2\rangle}(S)} \sum_{j\ne i} \sum_{x\in S_j} p_i^{\langle 2\rangle}(x)\|x-\mu_i\| &\le \frac{1}{p_i^{\langle 2\rangle}(S)} \sum_{j\ne i} \sum_{x\in S_j} e^{-c_{ij}^2 \sigma_{ij}^2 d/8\sigma_i^2} \sqrt{(\sigma_j^2 + c_{ij}^2 \sigma_{ij}^2)(1+2\varepsilon_o)d} \\
&\le \frac{1}{p_i^{\langle 2\rangle}(S)} \sum_{x\notin S_i} e^{-c^2d/16} \sigma_i \sqrt{d} \\
&\le \frac{m}{|S_i| \cdot (1-ke^{-c^2d/8})} e^{-c^2d/16} \sigma_i \sqrt{d} \;\le\; \frac{3}{w_i} e^{-c^2d/16} \sigma_i \sqrt{d}.
\end{aligned}$$

For the very last bound we use $|S_i| \ge \frac{1}{2} m w_i$ (Lemma 4(e)). ∎

Here's a summary of everything we have so far.

Theorem 16 *Suppose that $l > k$, that $w_i \ge w_{min}$ for all i, and that conditions (C1)–(C4) hold. With probability at least $1 - 2m^2 e^{-\varepsilon_o^2 d/8} - 2m^2 k e^{-\varepsilon_o^2 d/2} - 2ke^{-lw_{min}/12} - k(l+1)e^{-lw_{min}}$, the center-estimates returned after two rounds of EM satisfy (for each i):*

$$\|\mu_i^{\langle 2\rangle} - \mu_i\| \;\le\; \|\text{mean}(S_i) - \mu_i\| + \frac{5}{w_{min}} e^{-c^2 d/16} \sigma_i \sqrt{d}.$$

Choosing $c^2 d \ge 16 \ln \frac{5}{\varepsilon w_{min}}$ and $\varepsilon_o = \frac{1}{96}\min(1, c^2)$ gives Theorem 1. We can also prove bounds on the final mixing weights and variance.

Theorem 17 *To the results of Theorem 16 it can be added that if $c^2 d \geq 16 \ln \frac{5}{\varepsilon w_{min}}$ (for some $\varepsilon > 0$), then for any i,*

$$\frac{|S_i|}{m} \cdot (1-\varepsilon) \;\leq\; w_i^{\langle 2\rangle} \;\leq\; \frac{|S_i|}{m} + \varepsilon$$

and

$$(1-\varepsilon)\mathrm{var}(S_i) - \frac{\|\mu_i^{\langle 2\rangle} - \mathrm{mean}(S_i)\|^2}{d} \;\leq\; \sigma_i^{\langle 2\rangle 2} \;\leq\; (1+\varepsilon)\mathrm{var}(S_i) + \varepsilon\sigma_i^2 + \frac{\varepsilon\|\mu_i - \mathrm{mean}(S_i)\|^2}{d}$$

(where $\mathrm{var}(S_i)$ *is the empirical variance of cluster S_i).*

Proof. The bounds on $w_i^{\langle 2\rangle}$ come directly from writing

$$w_i^{\langle 2\rangle} \;=\; \frac{1}{m}\sum_{x\in S} p_i^{\langle 2\rangle}(x) \;=\; \frac{1}{m}\left(\sum_{x\in S_i} p_i^{\langle 2\rangle}(x) + \sum_{x\notin S_i} p_i^{\langle 2\rangle}(x)\right)$$

and then using our old bounds $p_i^{\langle 2\rangle}(x) \geq 1 - ke^{-c^2d/8}$ for $x \in S_i$ and $p_i^{\langle 2\rangle}(x) \leq e^{-c^2d/8}$ for $x \notin S_i$ (Lemma 14).

For the variance, we again exploit the alternative formulation of Lemma 10, but this time in a slightly different way. Let $\mu = \mathrm{mean}(S_i)$.

$$\sigma_i^{\langle 2\rangle 2} \;=\; \frac{\sum_{x\in S} p_i^{\langle 2\rangle}\|x-\mu\|^2}{d\, p_i^{\langle 2\rangle}(S)} - \frac{\|\mu_i^{\langle 2\rangle} - \mu\|^2}{d}.$$

Thus:

$$\begin{aligned}
\sigma_i^{\langle 2\rangle 2} \;&\geq\; \frac{\sum_{x\in S_i}(1-ke^{-c^2d/8})\|x-\mu\|^2}{d(|S_i| + me^{-c^2d/8})} - \frac{\|\mu_i^{\langle 2\rangle} - \mu\|^2}{d} \\
&=\; \frac{\sum_{x\in S_i}\|x - \mathrm{mean}(S_i)\|^2}{d|S_i|} \cdot \frac{1-ke^{-c^2d/8}}{1+me^{-c^2d/8}/|S_i|} - \frac{\|\mu_i^{\langle 2\rangle} - \mu\|^2}{d},
\end{aligned}$$

the first term of which is recognizable as $\mathrm{var}(S_i)$. Similarly,

$$\begin{aligned}
\sigma_i^{\langle 2\rangle 2} \;&\leq\; \frac{1}{d|S_i|(1-ke^{-c^2d/8})}\left\{\sum_{x\in S_i}\|x-\mu\|^2 + \sum_{x\notin S_i} e^{-c^2d/8}\cdot 2(\|x-\mu_i\|^2 + \|\mu_i - \mu\|^2)\right\} \\
&\leq\; \frac{\mathrm{var}(S_i)}{1-ke^{-c^2d/8}} + \frac{2m}{|S_i|(1-ke^{-c^2d/8})}\left(e^{-c^2d/16}\sigma_i^2 + e^{-c^2d/8}\frac{\|\mu_i-\mu\|^2}{d}\right),
\end{aligned}$$

from which the claim follows directly. ▌

9. Concluding Remarks

This paper provides a principled basis for answering some important questions surrounding EM: how many clusters should be used, how the parameters ought to be initialized, and how pruning should be carried out. These results may be of interest to practitioners of EM.

But what about the claim that EM can be made to work in just two rounds? This requires what we call the

Strong Gaussian assumption. The data are i.i.d. samples from a true mixture of Gaussians.

This assumption is the standard setting for other theoretical results about EM, but is it reasonable to expect of real data sets? We recommend instead the

Weak Gaussian assumption. The data looks like it comes from a particular mixture of Gaussians in the following sense: for any sphere in $\mathbb{R}^d$, the fraction of the data that falls in the sphere is the expected fraction under the mixture distribution, $\pm\varepsilon_0$, where ε_0 is some term corresponding to sampling error and will typically be proportional to $m^{-1/2}$, where m is the number of samples. Some other concept class of polynomial VC dimension can be used in place of spheres.

The strong assumption immediately implies the weak assumption (with high probability) by a large deviation bound, since the concept class of spheres in $\mathbb{R}^d$ has VC dimension just $d+1$ (Dudley, 1979; Haussler, 1992). What kinds of conclusions can we draw from the strong assumption but not the weak one? Here is an example: "if two data points are drawn from $N(0,I_d)$ then with overwhelming probability they are separated by a distance of at least $\sqrt{d}$". The weak assumption does not support this; with just two samples, in fact, the sampling error is so high that it does not allow us to draw any non-trivial conclusions at all.

It is often argued that the Gaussian is the most natural model of a cluster because of the central limit theorem. However, central limit theorems, more specifically Berry-Esséen theorems (Feller, 1966), yield Gaussians in the sense of the weak assumption, not the strong one. For the same reason, the weak Gaussian assumption arises naturally when we take random projections of mixtures of product distributions (Diaconis and Freedman, 1984). Ideally therefore, we could provide performance guarantees for EM under just this condition. It might be possible to extend our analysis appropriately; for an example of what needs to be changed in the algorithm, consider that the weak assumption allows $\sqrt{m}$ data points to be placed arbitrarily in space, and therefore an outlier removal procedure is probably necessary.

Acknowledgments

The first author is indebted to Daniel Hsu for suggesting many simplifications to the analysis, to the reviewers for significantly improving the presentation, and to the NSF for support under grant IIS-0347646.

References

D. Achlioptas and F. McSherry. On spectral learning of mixtures of distributions. In *Proceedings of the 18th Conference on Learning Theory*, 458–469, 2005.

S. Arora and R. Kannan. Learning mixtures of separated nonspherical Gaussians. *Annals of Applied Probability*, 15(1A):69–92, 2005.

C. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, 1995.

S. Dasgupta. Learning mixtures of Gaussians. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, 634–644, 1999.

A. Dempster, N. Laird, and D. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *J. Royal Statist. Soc. Ser. B*, 39:1–38, 1977.

P. Diaconis and D. Freedman. Asymptotics of graphical projection pursuit. *Annals of Statistics*, 12:793–815, 1984.

R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. John Wiley, New York, 1973.

R. Dudley. Balls in R^k do not cut all subsets of $k+2$ points. *Advances in Mathematics*, 31:306–308, 1979.

R. Durrett. *Probability: Theory and Examples*. Duxbury, Belmont, California, 1996.

W. Feller. *An Introduction to Probability Theory and its Applications*, vol. II. John Wiley, New York, 1966.

T. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

D. Haussler. Decision-theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100:78–150, 1992.

M. Kearns, Y. Mansour, and A. Ng. An information-theoretic analysis of hard and soft assignment methods for clustering. In *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence*, 1997.

M. Kearns and U.V. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, Massachusetts, 1994.

R. Redner and H. Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195–239, 1984.

D. Titterington, A. Smith, and U. Makov. *Statistical Analysis of Finite Mixture Distributions*. John Wiley, London, 1985.

S. Vempala and G. Wang. A spectral algorithm for learning mixtures of distributions. *Journal of Computer and Systems Sciences*, 68(4):841–860, 2004.

S. Vempala, R. Kannan, and H. Salmasian. The spectral method for general mixture models. In *Proceedings of the 18th Conference on Learning Theory*, 2005.

C.F.J. Wu. On the convergence properties of the EM algorithm. *Annals of Statistics*, 11:95–103, 1983.

L. Xu and M.I. Jordan. On convergence properties of the EM algorithm for Gaussian mixtures. *Neural Computation*, 8:129–151, 1996.

Journal of Machine Learning Research 8 (2007) 227-248 Submitted 11/05; Revised 10/06; Published 2/07

Noise Tolerant Variants of the Perceptron Algorithm

Roni Khardon RONI@CS.TUFTS.EDU
Gabriel Wachman GWACHM01@CS.TUFTS.EDU
Department of Computer Science, Tufts University
Medford, MA 02155, USA

Editor: Michael J. Collins

Abstract

A large number of variants of the Perceptron algorithm have been proposed and partially evaluated in recent work. One type of algorithm aims for noise tolerance by replacing the last hypothesis of the perceptron with another hypothesis or a vote among hypotheses. Another type simply adds a margin term to the perceptron in order to increase robustness and accuracy, as done in support vector machines. A third type borrows further from support vector machines and constrains the update function of the perceptron in ways that mimic soft-margin techniques. The performance of these algorithms, and the potential for combining different techniques, has not been studied in depth. This paper provides such an experimental study and reveals some interesting facts about the algorithms. In particular the perceptron with margin is an effective method for tolerating noise and stabilizing the algorithm. This is surprising since the margin in itself is not designed or used for noise tolerance, and there are no known guarantees for such performance. In most cases, similar performance is obtained by the voted-perceptron which has the advantage that it does not require parameter selection. Techniques using soft margin ideas are run-time intensive and do not give additional performance benefits. The results also highlight the difficulty with automatic parameter selection which is required with some of these variants.

Keywords: perceptron algorithm, on-line learning, noise tolerance, kernel methods

1. Introduction

The success of support vector machines (SVM) (Boser et al., 1992; Cristianini and Shawe-Taylor, 2000) has led to increasing interest in the perceptron algorithm. Like SVM, the perceptron algorithm has a linear threshold hypothesis and can be used with kernels, but unlike SVM, it is simple and efficient. Interestingly, despite a large number of theoretical developments, there is no result that explains why SVM performs better than perceptron, and similar convergence bounds exist for both (Graepel et al., 2000; Cesa-Bianchi et al., 2004). In practice, SVM is often observed to perform slightly better with significant cost in run time. Several on-line algorithms have been proposed which iteratively construct large margin hypotheses in the feature space, and therefore combine the advantages of large margin hypotheses with the efficiency of the perceptron algorithm. Other variants adapt the on-line algorithms to work in a batch setting choosing a more robust hypothesis to be used instead of the last hypothesis from the on-line session. There is no clear study in the literature, however, that compares the performance of these variants or the possibility of combining them to obtain further performance improvements. We believe that this is important since these algorithms have already been used in applications with large data sets (e.g., Collins, 2002; Li et al., 2002) and a better understanding of what works and when can have a direct implication for future use. This paper provides such an experimental study where we focus on noisy data and more generally the "unrealizable case" where the data is simply not linearly separable. We chose some of the basic variants and experimented with them to explore their performance both with hindsight knowledge and in a statistically robust setting.

More concretely, we study two families of variants. The first explicitly uses the idea of hard and soft margin from SVM. The basic perceptron algorithm is mistake driven, that is, it only updates the hypothesis when it makes a mistake on the current example. The perceptron algorithm with margin (Krauth and Mézard, 1987; Li et al., 2002) forces the hypothesis to have some margin by making updates even when it does not make a mistake but where the margin is too small. Adding to this idea, one can mimic soft-margin versions of support vector machines within the perceptron algorithm that allow it to tolerate noisy data (e.g., Li et al., 2002; Kowalczyk et al., 2001). The algorithms that arise from this idea constrain the update function of the perceptron and limit the effect of any single example on the final hypothesis. A number of other variants in this family exist in the literature. Each of these performs margin based updates and has other small differences motivated by various considerations. We discuss these further in the concluding section of the paper.

The second family of variants tackles the use of on-line learning algorithms in a batch setting, where one trains the algorithm on a data set and tests its performance on a separate test set. In this case, since updates do not always improve the error rate of the hypothesis (e.g., in the noisy setting), the final hypothesis from the on-line session may not be the best to use. In particular, the longest survivor variant (Kearns et al., 1987; Gallant, 1990) picks the "best" hypothesis on the sequential training set. The voted perceptron variant (Freund and Schapire, 1999) takes a vote among hypotheses produced during training. Both of these have theoretical guarantees in the PAC learning model (Valiant, 1984). Again, other variants exist in the literature which modify the notion of "best" or the voting scheme among hypotheses and these are discussed in the concluding section of the paper.

It is clear that each member of the first family can be combined with each member of the second. In this paper we report on experiments with a large number of such variants that arise when combining some of margin, soft margin, and on-line to batch conversions. In addition to real world data, we used artificial data to check the performance in idealized situations across a spectrum of data types.

The experiments lead to the following conclusions: First, the perceptron with margin is the most successful variant. This is surprising since among the algorithms experimented with it is only one not designed for noise tolerance. Second, the soft-margin variants on their own are weaker than the perceptron with margin, and combining soft-margin with the regular margin variant does not provide additional improvements.

The third conclusion is that in most cases the voted perceptron performs similarly to the perceptron with margin. The voted perceptron has the advantage that it does not require parameter selection (for the margin) that can be costly in terms of run time. Combining the two to get the voted perceptron with margin has the potential for further improvements but this occasionally degrades performance. Finally, both the voted perceptron and the margin variant reduce the deviation in accuracy in addition to improving the accuracy. This is an important property that adds to the stability and robustness of the algorithms.

The rest of the paper is organized as follows. The next section reviews all the algorithms and our basic settings for them. Section 3 describes the experimental evaluation. We performed two kinds of experiments. In "parameter search" we report the best results obtained with any parameter setting.

```
Input set of examples and their labels $\mathcal{Z} = ((x_1,y_1),\ldots,(x_m,y_m)) \in (\mathbb{R}^n \times \{-1,1\})^m$,
$\eta$, $\theta_{Init}$, $C$

  • Initialize $w \leftarrow 0^n$ and $\theta \leftarrow \theta_{Init}$
  • for every training epoch:
  • for every $x_j \in \mathcal{X}$:
      – $\hat{y} \leftarrow sign(\langle w, x_j \rangle - \theta)$
      – if $(\hat{y} \neq y_j)$
          * $w \leftarrow w + \eta y_j x_j$
          * $\theta \leftarrow \theta + \eta y_j C$
```

Figure 1: The Basic Perceptron Algorithm

This helps set the scope and evaluate the potential of different algorithms to improve performance and provides insight about their performance, but it does not give statistically reliable results. In "parameter optimization" the algorithms automatically select the parameters and the performance can be interpreted statistically. The concluding section further discusses the results and puts related work in their context leading to directions for future work.

2. Algorithms

This section provides details of all the algorithms used in the experiments and our basic settings for them.

2.1 The Perceptron Algorithm

The perceptron algorithm (Rosenblatt, 1958) takes as input a set of training examples in $\mathbb{R}^n$ with labels in $\{-1,1\}$. Using a weight vector, $w \in \mathbb{R}^n$, initialized to 0^n, and a threshold, θ, it predicts the label of each training example x to be $y = sign(\langle w,x \rangle - \theta)$. The algorithm adjusts w and θ on each misclassified example by an additive factor. The algorithm is summarized in Figure 1 following the form by Cristianini and Shawe-Taylor (2000). Unlike Cristianini and Shawe-Taylor (2000), we allow for a non-zero value of θ_{Init} and we allow C to be initialized to any value.

The "learning rate" η controls the extent to which w can change on a single update. Note that if θ is initialized to 0 then η has no effect since $sign(\langle w,x_i \rangle - \theta) = y_i$ iff $sign(\eta(\langle w,x_i \rangle - \theta)) = y_i$. However, the initial choice of θ is important as it can be significant in early iterations of the algorithm. Note also that there is only one "degree of freedom" between η and θ_{Init} since scaling both by the same factor does not alter the result. So one could fix $\eta = 1$ and vary only θ_{Init}. But we keep the general form for convenience.

When the data are linearly separable via some hyperplane (w,θ), the margin is defined as $\gamma = \min_{1 \leq i \leq m}(y_i(\langle w,x_i \rangle - \theta))$. When $\|w\| = 1$, γ is the minimum Euclidean distance of any point in the data set to (w,θ). If the data are linearly separable, and θ is initialized to 0, the perceptron algorithm is guaranteed to converge in $\leq (\frac{R}{\gamma})^2$ iterations (Novikoff, 1962; Cristianini and Shawe-Taylor, 2000), where $R = \max_{1\leq i\leq m} \|x_i\|$.

In the case of non-separable data, the extent to which a data point fails to have margin γ via the hyperplane w can be quantified by a slack variable $\xi_i = \max(0, \gamma - y_i(\langle w, x_i\rangle + \theta))$. Observe that when $\xi_i = 0$, the example x_i has margin at least γ via the hyperplane defined by (w, θ). The perceptron is guaranteed to make no more than $(\frac{2(R+D)}{\gamma})^2$ mistakes on m examples, for any $w, \gamma > 0$ where $D = \sqrt{\sum_{i=1}^m \xi_i^2}$ (Freund and Schapire, 1999; Shalev-Shwartz and Singer, 2005). Thus one can expect some robustness to noise.

It is well known that the perceptron can be re-written in "dual form" whereby it can be used with kernel functions (Cristianini and Shawe-Taylor, 2000). The dual form of the perceptron is obtained by observing that the weight vector can be written as $w = \sum_{i=1}^m \eta\alpha_i y_i x_i$ where α_i is the number of mistakes that have been made on example i. Subsequently, a new example x_j is classified according to $sign(SUM - \theta)$ where $SUM = \sum_i \eta\alpha_i y_i \langle x_i, x_j\rangle$. Replacing the inner product with a kernel function yields the kernel perceptron. All the perceptron variants we discuss below have dual form representations. However, the focus of this paper is on noise tolerance and this is independent of the use of primal or dual forms and the use of kernels. We therefore present all the algorithms in primal form.

2.2 The λ-trick

The λ-trick (Kowalczyk et al., 2001; Li et al., 2002) attempts to minimize the effect of noisy examples during training similar to the L_2 soft margin technique used with support vector machines (Cristianini and Shawe-Taylor, 2000). We classify example x_j according to $sign(SUM - \theta)$ where

$$SUM = \langle w, x_j\rangle + I_j y_j \lambda \|x_j\|^2 \tag{1}$$

and where I_j is an indicator variable such that

$$I_j = \begin{cases} 1 & \text{if } x_j \text{ was previously classified incorrectly} \\ 0 & \text{otherwise} \end{cases}.$$

Thus during training, if a mistake has been made on x_j then in future iterations we increase SUM artificially by a factor of $\lambda\|x_j\|^2$ when classifying x_j but not when classifying other examples. A high value of λ can make the term $y_j\lambda\|x_j\|^2$ dominate the sum and make sure that x_j does not lead to an update, so it is effectively ignored. In this way λ can prevent large number of updates on noisy (as well as other) examples limiting their effect.

We note that this technique is typically presented using an additive λ instead of $\lambda\|x_j\|^2$. While there is no fundamental difference, adding $\lambda\|x_j\|^2$ is more convenient in the experiments since it allows us to use the same values of λ for all data sets (since the added term is scaled relative to the data).

2.3 The α-bound

This variant is motivated by the L_1 soft margin technique used with support vector machines (Cristianini and Shawe-Taylor, 2000). The α-*bound* places a bound α on the number of mistakes the algorithm can make on a particular example, such that when the algorithm makes a mistake on some x_i, it does not update w if more than α mistakes have already been made on x_i. As in the case of the λ-trick, the idea behind this procedure is to limit the influence of any particular noisy example on the hypothesis. Intuitively, a good setting for α is some fraction of the number of training iterations. To see that, assume that the algorithm has made α mistakes on a particular noisy example x_k. In all subsequent training iterations, x_k never forces an update, whereas the algorithm may continue to increase the influence of other non-noisy examples. If the ratio of non-noisy examples to noisy examples is high enough the algorithm should be able to bound the effect of noisy examples in the early training iterations while leaving sufficient un-bounded non-noisy examples to form a good hypothesis in subsequent iterations. While this is a natural variant, we are not aware of any experimental results using it. Note that in order for the α-bound to work effectively, the algorithm must perform a high enough number of training iterations.

2.4 Perceptron Using Margins

The classical perceptron attempts to separate the data but has no guarantees on the separation margin obtained. The Perceptron Algorithm using Margins (PAM) (Krauth and Mézard, 1987) attempts to establish such a margin, τ, during the training process. Following work on support vector machines (Boser et al., 1992; Cristianini and Shawe-Taylor, 2000) one may expect that providing the perceptron with higher margin will add to the stability and accuracy of the hypothesis produced.

To establish the margin, instead of only updating on examples for which the classifier makes a mistake, PAM updates on x_j if

$$y_j(SUM - \theta) < \tau$$

where SUM is as in Equation (1). Notice that this includes the case of a mistake where $y_j(SUM - \theta) < 0$ and the case of correct classification with low margin when $0 \leq y_j(SUM - \theta) < \tau$. In this way, the algorithm "establishes" some minimal separation of the data. Notice that the weight vector w is not normalized during the learning process and its norm can increase with updates. Therefore, the constraint imposed by τ becomes less restrictive as learning progresses, and the normalized margin is not τ. When the data are linearly separable and $\tau < \gamma_{opt}$, PAM finds a separating hyperplane with a margin that is guaranteed to be at least $\gamma_{opt}\frac{\tau}{\sqrt{8(\eta R^2+\tau)}}$, where γ_{opt} is the maximum possible margin (Krauth and Mézard, 1987; Li et al., 2002).[1]

As above, it is convenient to make the margin parameter data set independent so we can use the same values across data sets. To facilitate this we replace the above and update if

$$y_j(SUM - \theta) < \tau\theta_{Init}$$

so that τ can be thought of as measured in units of θ_{Init}.

A variant of PAM exists in which a different value of τ is chosen for positive examples than for negative examples (Li et al., 2002). This seems to be important when the class distribution is skewed. We do not study that variant in this paper.

2.5 Longest Survivor and Voted Perceptron

The classical perceptron returns the last weight vector w, that is, the one obtained after all training has been completed, but this may not always be useful especially if the data is noisy. This is a general issue that has been studied in the context of using on-line algorithms that expect one example at a

1. Other variants (e.g., Gentile, 2001; Tsampouka and Shawe-Taylor, 2005; Shalev-Shwartz and Singer, 2005) do normalize the weight vector and thus have better margin guarantees.

time in a batch setting where a set of examples is given for training and one hypothesis is used at the end to classify all future instances. Several variants exist that handle this issue. In particular Kearns et al. (1987) show that *longest survivor* hypothesis, that is, the one who made the largest number of correct predictions during training in the on-line setting, is a good choice in the sense that one can provide guarantees on its performance in the PAC learning model (Valiant, 1984). Several variations of this idea were independently proposed under the name of the *pocket algorithm* and empirical evidence for their usefulness was provided (Gallant, 1990).

The voted perceptron (Freund and Schapire, 1999) assigns each vector a "vote" based on the number of sequential correct classifications by that weight vector. Whenever an example is misclassified, the voted perceptron records the number of correct classifications made since the previous misclassification, assigns this number to the current weight vector's vote, saves the current weight vector, and then updates as normal. After training, all the saved vectors are used to classify future examples and their classifications are combined using their votes. At first sight the voted-perceptron seems to require expensive calculation for prediction. But as pointed out by Freund and Schapire (1999), the output of the weight vector resulting from the first k mistakes can be calculated from the output of the weight vector resulting from the first $k-1$ mistakes in constant time. So when using the dual perceptron the prediction phase of the voted perceptron is not substantially more expensive than the prediction phase of the classical perceptron, although it is more expensive in the primal form.

When the data are linearly separable and given enough iterations, both these variants will converge to a hypothesis that is very close to the simple perceptron algorithm. The last hypothesis will predict correctly on all examples and indeed its vote will be the largest vote among all hypotheses used. When the data are not linearly separable the quality of hypothesis may fluctuate during training as noisy examples are encountered.

2.6 Algorithms Summary

Figure 2 summarizes the various algorithms and prediction strategies in primal form. The algorithms have corresponding dual forms and kernelized versions that we address in the experimental sections.

2.7 Multi-Class Data

Some of the data sets we experiment with have more than two labels. For these we used a standard solution as follows. For each label l, we train a classifier to separate examples labeled l (the positive examples) from other examples (all of which become negative examples). This can be done on-line where we train all the classifiers "in parallel". During testing we calculate the weight given by each classifier (before the sign function is applied) and choose the one maximizing the weight.

3. Experimental Evaluation

We ran two sets of experiments with the algorithms described above. In one set of experiments we searched through a pre-determined set of values of τ, α, and λ by running each of the classical, longest survivor, and voted perceptron using 10-fold cross-validation and parameterized by each value of τ, α, and λ as well as each combination of $\tau \times \alpha$ and $\tau \times \lambda$. This first set of experiments is called *parameter search*. The purpose of the parameter search experiments was to give us a comprehensive view of the effects of the various parameters. This can show whether a method has any

```
TRAINING: Input set of examples and their labels
$\mathcal{Z} = ((x_1,y_1),\ldots,(x_m,y_m)) \in (\mathbb{R}^n \times \{-1,1\})^m$, $\theta_{Init}$, $\eta$, $C$, $\alpha$-bound, $\lambda$, $\tau$
  • Initialize $\alpha \leftarrow 0^m$, $k \leftarrow 0$, $w_0 \leftarrow 0^n$, $tally \leftarrow 0$, $best\_tally \leftarrow 0$, $\theta_0 \leftarrow \theta_{Init}$, $\theta_{l.s.} \leftarrow \theta_{Init}$
  • for every training iteration
  • for every $z_j \in \mathcal{Z}$:
      – $SUM \leftarrow \langle w_k, x_j \rangle + I_{(\alpha_j \neq 0)}\ y_j \lambda \|x_j\|^2$
      – Predict:
          ∗ if $SUM < \theta_k - \tau$ then $\hat{y} = -1$
          ∗ else if $SUM > \theta_k + \tau$ then $\hat{y} = 1$
          ∗ else $\hat{y} = 0$ // This forces an update
      – if $(\hat{y} \neq y_j)$ AND $(\alpha_j < \alpha\text{-bound})$
          ∗ $w_{k+1} \leftarrow w_k + \eta y_j x_j$
          ∗ $\alpha_j \leftarrow \alpha_j + 1$
          ∗ Update ''mistakes'' data structure
          ∗ $\theta_{k+1} \leftarrow \theta_k + \eta y_j C$
          ∗ $vote_{k+1} \leftarrow 0$
          ∗ $k \leftarrow k+1$
          ∗ $tally \leftarrow 0$
      – else if $(\hat{y} = y_j)$
          ∗ $vote_k \leftarrow vote_k + 1$
          ∗ $tally \leftarrow tally + 1$
          ∗ if $(tally > best\_tally)$
              · $best\_tally \leftarrow tally$
              · $w_{l.s.} \leftarrow w_k$
              · $\theta_{l.s.} \leftarrow \theta_k$
PREDICTION: To predict on a new example $x_{m+1}$:
  • CLASSICAL:
      – $SUM \leftarrow \langle w_k, x_{m+1} \rangle$
      – $\hat{y} \leftarrow sign(SUM - \theta_k)$
  • LONGEST SURVIVOR:
      – $SUM \leftarrow \langle w_{l.s.}, x_{m+1} \rangle$
      – $\hat{y} \leftarrow sign(SUM - \theta_{l.s.})$
  • VOTED:
      – for $i \leftarrow 1$ to k
          ∗ $SUM \leftarrow \langle w_i, x_{m+1} \rangle$
          ∗ $\hat{y} \leftarrow sign(SUM - \theta_i)$
          ∗ VOTES $\leftarrow$ VOTES $+ vote_i\ \hat{y}$
      – $\hat{y}_{final} \leftarrow sign(\text{VOTES})$
```

Figure 2: Illustration of All Algorithm Variants

chance of improving performance since the experiments give us hindsight knowledge. The experiments can also show general patterns and trends in the parameter landscape again giving insight into the performance of the methods. Notice that parameter search cannot be used to indicate good values of parameters as this would be hand-tuning the algorithm based on the test data. However, it can guide in developing methods for automatic selection of parameters.

In the second set of experiments, we used the same set of values as in the first experiment, but using a method for automatic selection of parameters. In particular we used a "double cross validation" where in each fold of the cross validation (1) one uses parameter search on the training data only using another level of cross validation, (2) picks values of parameters based on this search, (3) trains on the complete training set for the fold using these values, and (4) evaluates on the test set. We refer to this second set of experiments as *parameter optimization*. This method is expensive to run as it requires running all combinations of parameter values in each internal fold of the outer cross validation. So if both validations use 10 folds then we run the algorithm 100 times per parameter setting. This sets a strong limitation on the number of parameter variations that can be tried. Nonetheless this is a rigorous method of parameter selection.

Both sets of experiments were run on randomly-generated artificial data and real-world data from the University of California at Irvine (UCI) repository and other sources. The purpose of the artificial data was to simulate an ideal environment that would accentuate the strengths and weaknesses of each algorithm variant. The other data sets explore the extent to which this behavior is exhibited in real world problems.

For further comparison, we also ran SVM on the data sets using the L_1-norm soft margin and L_2-norm soft margin. We used SVM Light (Joachims, 1999) and only ran *parameter search*.

3.1 Data Set Selection and Generation

We have experimented with 10 real world data sets of varying size and 150 artificial data sets composed of 600 examples each.

We first motivate the nature of artificial data used by discussing the following four idealized scenarios. The simplest, type 1, is linearly separable data with very small margin. Type 2 is linearly separable data with a larger, user-defined margin. For type 3 we first generate data as in type 1, and then add random class noise by randomly reversing the labels of a given fraction of examples. For type 4, we generate data as in type 2, and then add random class noise. One might expect that the basic perceptron algorithm will do fine on type 1 data, the perceptron with margin will do particularly well on type 2 data, that noise tolerant variants without margin will do well on type 3, and that some combination of noise tolerant variant with margin will be best for type 4 data. However, our experiments show that the picture is more complex; preliminary experiments confirmed that the expected behavior is observed, except that the perceptron with margin performed well on data of type 3 as well, that is, when the "natural margin" was small and the data was not separable due to noise. In the following, we report on experiments with artificial data where the margin and noise levels are varied so we effectively span all four different types.

Concretely the data was generated as follows. We first specify parameters for number of features, noise rate and the required margin size. Given these, each example x_i is generated independently and each attribute in an example is generated independently using an integer value in the range $[-10,10]$. We generated the weight vector, w, by flipping a coin for each example and adding it to the weight vector on a head. We chose θ as the average of $\frac{1}{2}|\langle w,x_i\rangle|$ over all x_i. We

Name	# features*	# examples	Baseline
Adult	105	32561	75.9
Breast-cancer-wisconsin	9	699	65.5
Bupa	6	345	58
USPS	256	9298	N/A
Wdbc	30	569	62.7
Crx	46	690	55.5
Ionosphere	34	351	64.4
Wpbc	33	198	76.3
sonar.all-data	60	208	53.4
MNIST	784	70,000	N/A
*after preprocessing			

Table 1: UCI and Other Natural Data Set Characteristics

measured the actual margin of the examples with respect to the weight vector and then discarded any examples x_i such that $|\langle w,x_i\rangle - \theta| < M * \theta$ where M is the margin parameter specified. For the noisy settings, for each example we randomly switched the label with probability equal to the desired noise rate. In the tables of results presented below, f stands for number of features, M stands for the margin size, and N stands for the noise rate. We generated data sets with parameters $(f,M,N) \in \{50,200,500\} \times \{0.05,0.1,0.25,0.5,0.75\} \times \{0,0.05,0.1,0.15,0.25\}$, and for each parameter setting we generated two data sets, for a total 150 data sets.

For real world data we first selected two-class data sets from the UCI Machine Learning Repository (Newman et al., 1998) that have been used in recent comparative studies or in recent papers on linear classifiers (Cohen, 1995; Dietterich et al., 1996; Dietterich, 2000; Garg and Roth, 2003). Since we require numerical attributes, any nominal attribute in these data sets was translated to a set of binary attributes each being an indicator function for one of the values. Since all these data sets have a relatively small number of examples we added three larger data sets to strengthen statistically our conclusions: "Adult" from UCI (Newman et al., 1998), and "MNIST[2]" and "USPS[3]," the 10-class character recognition data sets. Due to their size, however, for the "USPS," "MNIST," and "Adult" data sets, there is no outer cross-validation in *parameter optimization*. For ease of comparison to other published results, we use the 7291/2007 training/test split for "USPS", and a 60000/10000 split for "MNIST".

The data sets used and their characteristics after the nominal-to-binary feature transformation are summarized in Table 1. The column labeled "Baseline" indicates the percentage of the majority class.

3.2 Exploratory Experiments and General Setup

The algorithms described above have several parameters that can affect their performance dramatically. In this paper we are particularly interested in studying the effect of parameters related to noise tolerance, and therefore we fix the value of θ_{Init}, η, C, and the number of training iterations. Prior to

2. `http://yann.lecun.com/exdb/mnist/`.
3. `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html`.

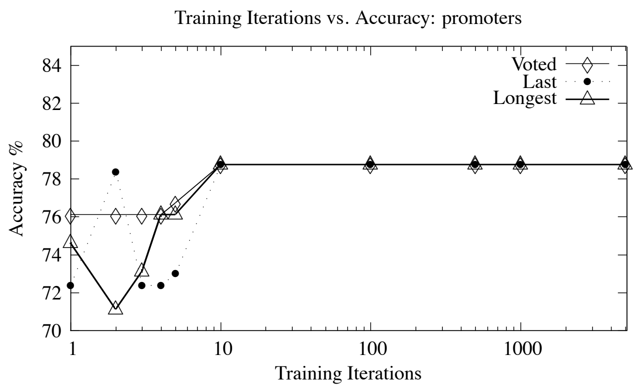


Figure 3: Example Learning Curve for UCI Data Set

fixing these values, we ran preliminary experiments in which we varied these parameters. In addition we ran experiments where we normalized the example vectors. The results showed that while the performance of the algorithms overall was different in these settings, the relationship between the performance of individual algorithms seems to be stable across these variations.

We set $\theta_{Init} = avg(\langle x_i, x_i \rangle)$, initializing the threshold at the same scale of inner products. Combined with a choice of $\eta = 0.1$, this makes sure that a few iterations should be able to guarantee that an example is classified correctly given no other changes to the hypothesis. We also set $C = avg(\langle x_i, x_i \rangle) = \theta_{Init}$. This is similar to the version analyzed by Cristianini and Shawe-Taylor (2000) except that we use the average instead of maximum norm. Using this setting we essentially maintain θ in units of θ_{Init}.

As explained above, the number of iterations must be sufficiently high to allow the α parameter to be effective, as well as to allow the weight vector to achieve some measure of stability. Typically a small number of iterations is sufficient, and preliminary experiments illustrated by the curve in Figure 3 showed that 100 iterations are more than enough to allow all the algorithms to converge to a stable error rate. Therefore, except where noted below, we report results for 100 iterations.

For the large data sets, we reduced the number of training iterations and increased η accordingly in order to run the experiments in a reasonable amount of time; for "Adult" we set $\eta = 0.4$ with 5 training iterations and for "MNIST" we set $\eta = 1$ with 1 training iteration. For "USPS," we used $\eta = 0.1$ and 100 iterations for *parameter search*, and $\eta = 0.1$ and 10 iterations for *parameter optimization*.

The parameters of interest in our experiments are τ, α, λ that control the margin and soft margin variants. Notice that we presented these so that their values can be fixed in a way that is data set independent. We used values for $\tau \in \{0, 0.125, 0.25, 0.5, 1, 2, 4\}$, $\alpha \in \{\infty, 80, 60, 40, 20, 10, 5\}$, and $\lambda \in \{0, 0.125, 0.25, 0.5, 1, 2, 4\}$. Notice that the values as listed from left to right vary from no effect to a strong effect for each parameter. We ran *parameter search* and *parameter optimization* over all of these values, as well as all combinations $\tau \times \lambda$ and $\tau \times \alpha$. Since *parameter optimization* over combined values is particularly expensive we have also experimented with a variant that first searches for a good τ value and then searches for a value of α (denoted $\tau \rightarrow \alpha$) and a variant that does likewise with τ and λ (denoted $\tau \rightarrow \lambda$).

We did not perform any experiments involving α on "MNIST" or "Adult" as their size required too few iterations to justify any reasonable α-bound.

	V > C	V > LS	LS > C	LS > V	C > LS	C > V	V = LS = C
Noise = 0	0	0	0	0	0	0	30
Noise = 0.05	12	10	11	3	0	2	16
Noise = 0.1	15	15	14	3	3	3	12
Noise = 0.15	16	14	13	5	6	3	10
Noise = 0.25	16	12	13	8	7	4	10

Table 2: Noise Percentage vs. Dominance: V = Voted, C = Classical, LS = Longest Survivor

Finally we performed a comparison of the classical perceptron, the longest survivor and the voted perceptron algorithms without the additional variants. Table 2 shows a comparison of the accuracies obtained by the algorithms over the artificial data. We ignore actual values but only report whether one algorithm gives higher accuracy or whether they tie (the variance in actual results is quite large). One can see that with higher noise rate the voted perceptron and longest survivor improve the performance over the base algorithm. Over all 150 artificial data sets, the voted perceptron strictly improves performance over the classical perceptron in 59 cases, and ties or improves in 138 cases. Using the distribution over our artificial data sets one can calculate a simple weak statistical test that supports the hypothesis that using the voted algorithm does not hurt accuracy, and can often improve it.

Except where noted, all the results reported below give average accuracy in 10-fold cross-validation experiments. To avoid any ordering effects of the data, the training set is randomly permuted before running the experiments.

3.3 Parameter Search

The *parameter search* experiments reveal several interesting aspects. We observe that in general the variants are indeed helpful on the artificial data since the performance increases substantially from the basic version. The numerical results are shown in Tables 3 and 4 and discussed below. Before showing these we discuss the effects of single parameters. Figure 4 plots the effect of single parameters for several data sets. For the artificial data set shown, the accuracy is reasonably well-behaved with respect to the parameters and good performance is obtained in a non negligible region; this is typical of the results from the artificial data. Data obtained for the real world data sets show somewhat different characteristics. In some data sets little improvement is obtained with any variant or parameter setting. In others, improvement was obtained for some parameter values but the regions were not as large implying that that automatic parameter selection may not be easy. Nonetheless it appears that when improvement is possible, τ on it own was quite effective when used with the classical perceptron. Notice that τ is consistently effective across all data sets; λ and α do not help on all data sets and λ sometimes leads to a drop in performance. As one might expect, our preliminary experiments also showed that very large values of τ harm performance significantly. These are not shown in the graphs as we have limited the range of τ in the experiments.

Tables 3 and 4 summarize the results of parameter search experiments on the real world data and some of the artificial data, respectively. For each data set and algorithm the tables give the best accuracy that can be achieved with parameters in the range tested. This is useful as it can indicate whether an algorithm has a potential for improvement, in that for some parameter setting it gives good performance. In order to clarify the contribution of different parameters, each column with

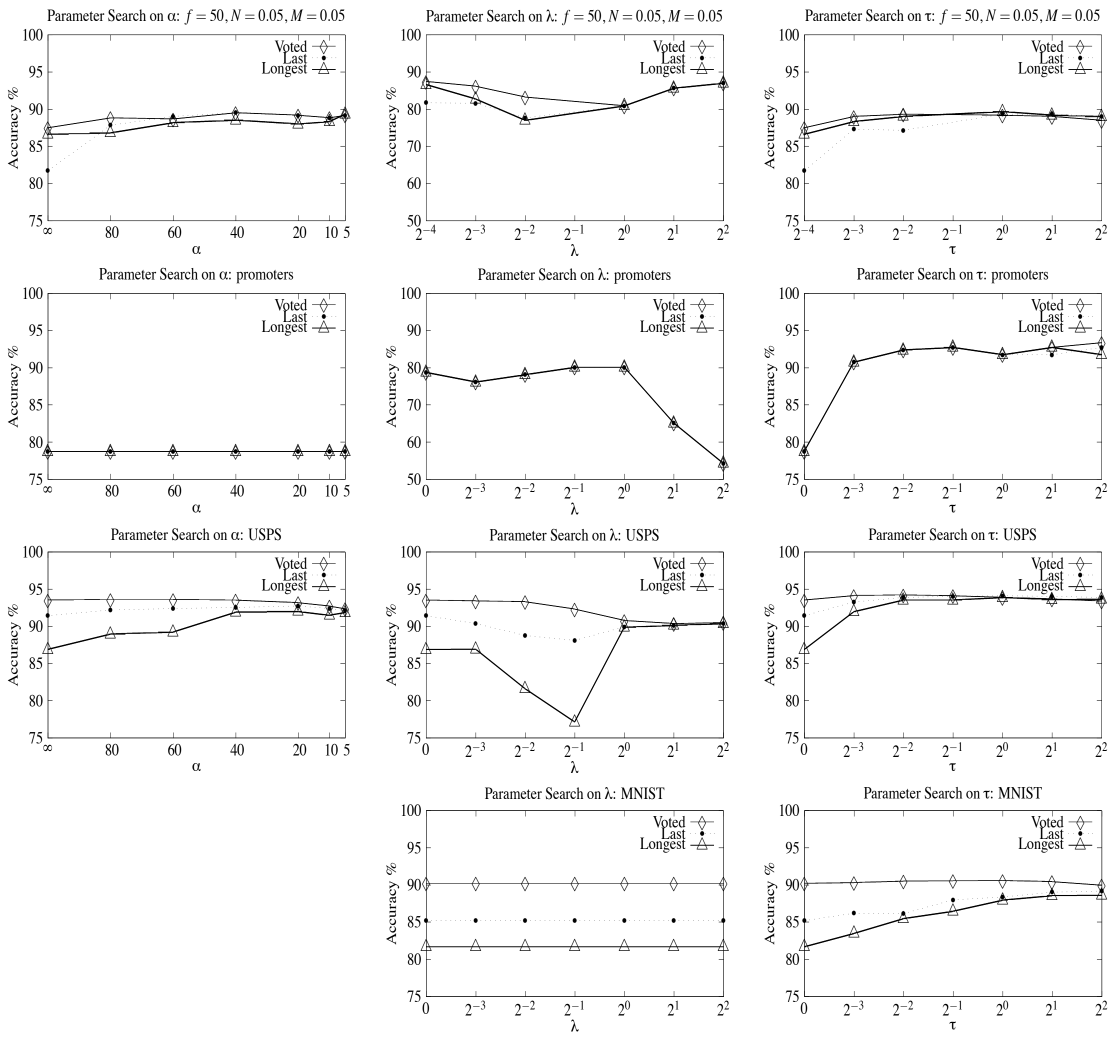


Figure 4: Parameter Search on Artificial and Real-world Data

parameters among τ, α, λ includes all values for the parameter except the non-active value. For example, any accuracy obtained in the τ column is necessarily obtained with a value $\tau \neq 0$. The column labeled "Nothing" shows results when all parameters are inactive.

Several things can be observed in the tables. Consider first the margin parameters. We can see that τ is useful even in data sets with noise; this is obvious both in the noisy artificial data sets and in the real-world data sets, all of which are inseparable in the native feature space. We can also see that α and λ do improve performance in a number of cases. However, they are less effective in general than τ, and do not provide additional improvement when combined with τ.

Looking next at the on-line to batch conversions we see that the differences between the basic algorithm, the longest survivor and the voted perceptron are noticeable without margin based variants. For the artificial data sets this only holds for one group of data sets ($f = 50$), the one with

		Baseline	Nothing	τ	λ	α	$\tau \times \lambda$	$\tau \times \alpha$
breast-cancer-wisconsin	Last	65.5	90.6	96.8	97.2	96.9	97.3	97.2
	Longest		96.9	97.0	97.2	97.0	97.3	97.2
	Voted		96.9	96.8	97.2	96.9	97.3	97.2
bupa	Last	58	57.5	71.8	64.1	69.2	71.5	67.8
	Longest		64.1	58.2	64.8	68.4	60.9	61.2
	Voted		68.9	65.9	67.7	70.7	67.4	66.0
wdbc	Last	62.7	92.4	93.2	92.8	93.3	93.9	92.6
	Longest		92.4	93.2	92.6	92.4	92.8	92.8
	Voted		92.3	92.0	93.2	92.4	93.2	92.4
crx	Last	55.5	62.5	68.6	65.5	66.7	69.0	66.7
	Longest		55.1	63.6	65.5	63.5	65.5	65.2
	Voted		64.9	65.4	65.5	66.7	65.5	66.4
promoters	Last	50	78.8	92.8	82.1	78.8	94.4	93.8
	Longest		78.8	92.8	82.1	78.8	94.4	94.4
	Voted		78.8	93.4	82.1	78.8	94.4	93.8
ionosphere	Last	64.4	86.6	87.5	86.9	87.7	88.6	87.2
	Longest		87.2	87.5	87.8	88.0	88.3	87.7
	Voted		88.0	87.7	87.5	88.6	88.9	87.5
wpbc	Last	76.3	77.3	76.4	80.6	76.9	79.0	76.4
	Longest		77.2	76.4	77.4	78.5	76.9	76.4
	Voted		78.8	76.4	80.6	77.4	78.5	76.4
sonar.all-data	Last	53.4	71.9	74.6	72.5	77.1	75.8	79.1
	Longest		75.3	77.1	73.3	77.2	77.7	78.7
	Voted		75.1	77.2	73.8	77.1	78.7	77.7
USPS	Last	N/A	91.5	94.1	90.4	92.8	94.3	94.1
	Longest	N/A	86.9	93.9	90.4	92	93.9	93.9
	Voted	N/A	93.5	94.2	93.4	93.6	94.3	94.3
MNIST	Last	N/A	85.2	89.2	85.2		89.2	
	Longest	N/A	81.7	88.6	81.7		88.6	
	Voted	N/A	90.2	90.6	90.2		90.6	

Table 3: Parameter Search on UCI and Other Data Sets

highest ratio of number of examples to number of features (12 : 1).[4] The longest survivor seems less stable and degrades performance in some cases.

Finally compare the τ variant with voted perceptron and their combination. For the artificial data sets using τ alone (with last hypothesis) gives higher accuracy than using the voted perceptron alone. For the real-world data sets this trend is less pronounced. Concerning their combination we see that while τ always helps the last hypothesis, it only occasionally helps voted, and sometimes hurts it.

Table 5 gives detailed results for parameter search over τ on the adult data set, where we also parameterize the results by the number of examples in the training set. We see that voted perceptron and the τ variant give similar performance on this data set as well. We also see that that in contrast

4. The results for data with $f = \{200, 500\}$ are omitted from the table. In these cases these was no difference between the basic, longest and voted versions except when combining with other variants.

Noise Pctg. (N)	Nothing	τ	λ	α	$\tau \times \lambda$	$\tau \times \alpha$
Last, $N = 0$	95.4	97	95.6	96.1	96.8	97
Longest	95.4	96.6	95.6	95.9	97.3	96.8
Voted	95.4	96.6	95.4	96.1	96.8	97
Last, $N = 0.05$	81.7	89.4	87	89.5	89.7	89.9
Longest	86.6	89.7	87	89.3	89.9	89.9
Voted	87.5	89.4	87	89.5	89.7	90.2
Last, $N = 0.1$	77.8	84.6	81.9	85.1	84.9	85.8
Longest	77.2	84.9	81.9	85.1	84.6	85.5
Voted	81.6	84.9	81.9	85.1	84.9	85.6
Last, $N = 0.15$	71.2	81.6	79.9	80.6	81.6	81.7
Longest	72.9	80.7	79.9	79.2	80.7	81.7
Voted	75.6	81.6	79.9	80.6	82.1	81.6
Last, $N = 0.25$	59.9	70.7	68.2	70.7	71.1	71.1
Longest	65	70.7	68.2	70.1	70.4	70.6
Voted	68	70.4	69.4	70.7	70.6	71.1

Table 4: Parameter Search on Artificial Data Sets with $f = 50$ and $M = 0.05$

with the performance on the artificial data mentioned above, voted perceptron performs well even with small training set size (and small ratio of examples to features).

The table adds another important observation about stability of the algorithms. Note that since we report results for concrete values of τ we can measure the standard deviation in accuracy observed. One can see that both the τ variant and the voted perceptron significantly reduce the variance in results. The longest survivor does so most of the time but not always. The fact that the variants lead to more stable results is also consistently true across the artificial and UCI data sets discussed above and is an important feature of these algorithms.

Finally, recall that the *average algorithm* (Servedio, 1999), which updates exactly once on each example, is known to tolerate classification noise for data distributed uniformly on the sphere and where the threshold is zero. For comparison, we ran this algorithm on all the data reflected in the tables above and it was not competitive. Cursory experiments to investigate the differences show that, when the data has a zero threshold separator and when perceptron is run for just one iteration, then (as reported in Servedio, 1999) the average algorithm performs better than basic perceptron. However, the margin based perceptron performs better and this becomes more pronounced with non-zero threshold and multiple iterations. Thus in some sense the perceptron variants are doing something non-trivial that is beyond the scope of the simple average algorithm.

3.4 Parameter Optimization

We have run the parameter optimization on the real-world data sets and the artificial data sets. Selected results are given in Tables 6 and 7. For these experiments we report average accuracy across the outer cross-validation as well as a 95% T-confidence interval around these, as suggested

# examples:	10		100		1000		5000		10000		20000	
	Accuracy	Std. Dev.										
$\tau = 0$												
Last	75.6	3.5	70.2	9.2	77.8	3.1	78.3	4.1	80.3	2.8	76.5	11.1
Longest	75.7	3.5	76.5	3.6	79.9	3.5	81.8	1.8	82.9	0.7	82.3	1.3
Voted	75.8	3.4	79.4	1.8	82.5	0.5	84	0.7	84.2	0.6	84.5	0.5
$\tau = 0.125$												
Last	72.7	4.6	73.3	8.9	79.7	3.7	80.8	2.7	82.7	1.4	80.4	3.9
Longest	73.1	4.6	76.6	3.9	81	2.1	81.1	2.4	81.8	1.5	82.1	1.7
Voted	74.1	4.4	79.6	1.4	82.7	0.5	83.9	0.7	84.2	0.6	84.3	0.4
$\tau = 0.25$												
Last	74.5	1.8	75.1	7.4	79.3	5.1	80.5	2.8	81.5	2.2	80.6	3.7
Longest	73.9	3.9	77.2	2.1	80.3	2.6	81.8	1.8	81.7	1.7	82.3	2.4
Voted	74.1	4.7	79.5	1.5	82.7	0.5	83.8	0.7	84.1	0.7	84.3	0.5
$\tau = 0.5$												
Last	71.8	6.3	75.5	1.9	80.4	3.4	82.6	1	82.7	1	82.8	1.5
Longest	74	5.7	77.2	7.5	80.4	2.3	82.1	1.6	82.5	1.6	80.8	2.9
Voted	73.6	5.6	78.3	0.8	82.6	0.6	83.7	0.7	84	0.7	84.2	0.5
$\tau = 1$												
Last	72.9	6.6	78.4	2.5	81.9	1.2	83	1.1	82.7	1.5	83.5	0.8
Longest	75.9	0.3	75.9	0.9	78.3	2.8	82.5	1.1	83	1	81.9	2.7
Voted	74.8	3.4	76.4	0.9	82.4	0.6	83.5	0.7	83.7	0.7	84.1	0.5
$\tau = 2$												
Last	75.5	1.2	75.8	2.6	82.4	0.4	83.2	0.9	82.8	1.2	83.6	0.8
Longest	70.7	15.6	75.9	0.9	77.8	2.7	81.4	2.9	82.5	2	81.9	2.8
Voted	70.7	15.6	76.4	1.8	81.8	1	83.2	0.7	83.5	0.6	83.8	0.6
$\tau = 4$												
Last	75.9	0.3	75.7	1.4	81.8	1	83.1	2.1	83.3	0.6	83.6	0.6
Longest	70.7	15.6	75.9	0.9	75.9	0.5	81.8	0.7	79.4	3.4	80.8	3.1
Voted	70.7	15.6	75.9	0.9	78.4	2.1	83	0.7	83.2	0.6	83.5	0.6

Table 5: Performance on “Adult” Data Set as a Function of Margin and Training Set Size

in Mitchell (1997).[5] No confidence interval is given for “USPS,” “MNIST,” or “Adult,” as the outer cross-validation loop is not performed.

In contrast with the tables for parameter search, columns of parameter variants in Tables 6 and 7 do include the inactive option. Hence the search in the τ column includes the value of $\tau = 0$ as well. This makes sense in the context of parameter optimization since the algorithm can choose between different active values and the inactive value. Notice that the standard deviation in accuracies is high on these data sets. This highlights the difficulty of parameter selection and algorithm comparison and suggests that results from single split into training and test sets that appear in the literature may not be reliable.

5. In more detail, we use the maximum likelihood estimate of the standard deviation σ, $s = \sqrt{\frac{1}{k}\sum_i (y_i - \bar{y})^2}$ where k is the number of folds (here $k = 10$), y_i is the accuracy estimate in each fold and $\bar{y}$ is the average accuracy. We then use the T confidence interval $y \in \hat{y} \pm t_{(k-1),0.975}\sqrt{\frac{1}{k(k-1)}\sum(y_i - \bar{y})^2}$. Notice that since $t_{9,0.975} = 2.262$ and $k = 10$ the confidence interval is 0.754 of the standard deviation.

	Nothing	τ	λ	α	$\tau \times \alpha$	$\tau \times \lambda$	$\tau \rightarrow \alpha$	$\tau \rightarrow \lambda$	SVM L1	SVM L2
Breast-cancer-wisconsin										
Last	90.6 +/- 2.3	95.2 +/- 2.1	95.2 +/- 3.1	94.7 +/- 3.3	95.3 +/- 2.6	96.3 +/- 2.1	95.1 +/- 2.3	96.9 +/- 1.3		
Longest	96.9 +/- 1.2	96.8 +/- 1.7	97.2 +/- 1.4	96.3 +/- 1.9	96.8 +/- 1.6	97 +/- 1.6	96.8 +/- 1.7	96.9 +/- 1.8	96.8 +/- 2.2	96.7 +/- 1.7
Voted	96.9 +/- 1.3	96.8 +/- 1.4	97 +/- 1.4	96.6 +/- 1.6	97 +/- 1.6	97.2 +/- 1.4	96.8 +/- 1.4	96.9 +/- 1.5		
Bupa										
Last	57.5 +/- 7.8	69.8 +/- 6.5	61.5 +/- 6.3	68.9 +/- 4.1	69.9 +/- 5.7	69.8 +/- 5.8	69.8 +/- 6.5	70.9 +/- 5.9		
Longest	64.1 +/- 7.1	64.1 +/- 7.1	64.1 +/- 6.6	65.3 +/- 4.3	65.3 +/- 4.3	64.1 +/- 6.6	65.3 +/- 4.3	64.1 +/- 6.6	66.5 +/- 8.6	63.2 +/- 5.2
Voted	68.9 +/- 5.8	68.9 +/- 5.8	67.5 +/- 6.4	68.9 +/- 6.3	68.9 +/- 6.3	67.5 +/- 6.2	68.9 +/- 6.3	67.2 +/- 6.0		
Wdbc										
Last	92.4 +/- 2.9	93 +/- 2.7	93.2 +/- 2.5	91.6 +/- 2.7	92.8 +/- 2.8	93.2 +/- 2.8	93 +/- 2.7	93.5 +/- 2.2		
Longest	92.4 +/- 2.0	91.7 +/- 2.5	92.1 +/- 2.5	92.3 +/- 2.3	93.2 +/- 1.7	92.5 +/- 2.1	92.6 +/- 2.1	91.4 +/- 2.9	92.8 +/- 4.4	94.7 +/- 2.1
Voted	92.3 +/- 2.3	92.3 +/- 2.6	92.8 +/- 2.7	91.7 +/- 1.9	91.6 +/- 2.2	92.4 +/- 2.4	91.6 +/- 2.6	92.8 +/- 2.3		
Crx										
Last	62.5 +/- 5.1	68.7 +/- 2.9	64.5 +/- 4.1	65.8 +/- 4.1	68.1 +/- 3.5	68.7 +/- 2.2	68.7 +/- 2.9	67.8 +/- 2.8		
Longest	55.1 +/- 7.3	60.4 +/- 6.5	62.6 +/- 4.8	62.6 +/- 4.4	64.5 +/- 4.9	60.9 +/- 5.7	62.9 +/- 4.6	59.4 +/- 6.7	76.6 +/- 13.7	65.9 +/- 22.8
Voted	64.9 +/- 4.0	64.2 +/- 2.7	65.4 +/- 3.2	66.2 +/- 3.8	66.4 +/- 3.7	64.9 +/- 2.6	65.7 +/- 3.3	64.3 +/- 3.1		
Ionosphere										
Last	86.6 +/- 3.8	87.2 +/- 3.4	86.3 +/- 3.5	85.7 +/- 4.8	86.9 +/- 3.4	86.9 +/- 2.6	86.6 +/- 3.0	86.6 +/- 2.6		
Longest	87.2 +/- 3.8	86.9 +/- 2.6	87.8 +/- 3.6	87.5 +/- 3.5	86.9 +/- 4.0	87.2 +/- 3.2	86.9 +/- 3.4	87.7 +/- 2.9	87.1 +/- 7.1	86.9 +/- 4.2
Voted	88 +/- 3.7	86.3 +/- 4.3	86.6 +/- 3.5	87.7 +/- 3.2	87.5 +/- 3.2	87.7 +/- 3.1	86 +/- 4.9	86 +/- 3.9		
Wpbc										
Last	77.3 +/- 4.7	76.7 +/- 4.4	76.4 +/- 4.4	75.1 +/- 6.1	75.1 +/- 6.1	77 +/- 5.2	75.7 +/- 6.2	78.3 +/- 5.1		
Longest	77.2 +/- 3.8	76.7 +/- 5.3	75.8 +/- 3.6	75.8 +/- 6.0	76.9 +/- 4.4	74.8 +/- 4.3	75.8 +/- 4.4	74.6 +/- 5.4	78.6 +/- 7.8	78.6 +/- 8.4
Voted	78.8 +/- 4.7	78.5 +/- 4.8	79 +/- 5.0	76.4 +/- 4.8	76.9 +/- 4.8	79 +/- 5.0	76.9 +/- 4.8	78.5 +/- 5.1		
Sonar										
Last	71.9 +/- 6.3	73.1 +/- 5.4	72.4 +/- 6.0	75.5 +/- 6.4	72.1 +/- 7.4	71.1 +/- 6.5	74.1 +/- 6.4	73.6 +/- 5.1		
Longest	75.3 +/- 5.1	77.6 +/- 6.2	73.3 +/- 6.0	74.3 +/- 6.9	73.5 +/- 5.9	74.1 +/- 7.5	74 +/- 5.5	78.1 +/- 6.6	57.2 +/- 11.9	55 +/- 11.8
Voted	75.1 +/- 6.0	75.6 +/- 6.9	74.3 +/- 5.6	77.5 +/- 7.0	78.1 +/- 7.0	77.1 +/- 7.2	76.1 +/- 7.9	75.6 +/- 6.9		
f=50,N=0.05,M=0.05										
Last	81.7 +/- 6.5	87.7 +/- 6.1	84.6 +/- 6.6	87.2 +/- 5.2	88 +/- 5.2	89.4 +/- 4	89 +/- 4.6	89.5 +/- 4.2		
Longest	86.6 +/- 4.3	88.2 +/- 4.2	85.6 +/- 4.7	87.3 +/- 6.1	89.4 +/- 4.5	88.2 +/- 3.9	89 +/- 5.4	88.8 +/- 3.6		
Voted	87.5 +/- 3.3	88.8 +/- 4.8	86.3 +/- 3.9	88.3 +/- 4.2	88.2 +/- 4.9	88.8 +/- 4.8	88.7 +/- 4.9	88.8 +/- 4.8		

Table 6: Parameter Optimization Results for UCI and Artificial Data Sets

	Nothing	τ	λ	$\tau \times \lambda$	$\tau \to \lambda$
USPS					
Last	87.4	90.7	87.4	90.3	90.2
Longest	87.5	89.9	87.4	90.3	89.9
Voted	89.7	90.9	89.6	90.9	90.9
Adult					
Last	81.9	83.9	83	83.8	83.8
Longest	83.4	83.4	83.4	83.6	83.4
Voted	84.4	84.2	84.4	84.3	84.3
MNIST					
Last	85.6	87.1	85.5	87.1	87.1
Longest	79.8	86.8	79.8	86.8	86.8
Voted	88	88.4	88	88.4	88.4
USPS,degree 4 poly kernel					
Last	92.9	93.6			
Longest	91.6	92.9			
Voted	92.7	93.2			
MNIST,degree 4 poly kernel					
Last	95.2	95.4			
Longest	93.2	94.9			
Voted	94.8	95			

Table 7: Parameter Optimization Results for Large Data Sets

Table 6 shows improvement over the basic algorithm in all data sets where parameter search suggested a potential for improvement, and no decrease in performance in the other cases, so the parameter selection indeed picks good values. Both τ and the voted perceptron provide consistent improvement over the classical perceptron; the longest survivor provides improvement over the classical perceptron on its own, but a smaller one than voted or τ in most cases. Except for improvements over the classical perceptron, none of the differences between algorithms is significant according to the T-intervals calculated. As observed above in *parameter search*, the variants with α and λ offer improvements in some cases, but when they do, τ and voted almost always offer a better improvement. Ignoring the intervals we also see that neither the τ variant nor the voted perceptron dominates the other. Combining the two is sometimes better but may decrease performance in the high variance cases.

For further comparison we also ran experiments with kernel based versions of the algorithms. Typical results in the literature use higher degree polynomial kernel on the “MNIST” and “USPS” data sets. Table 7 includes results using τ with a degree 4 polynomial kernel for these data sets. We can see that for “MNIST” the variants make little difference in performance but that for “USPS” we get small improvements and the usual pattern relating the variants is observed.

Finally, Table 6 also gives results for SVM. We have used SVMlight (Joachims, 1999) and ran with several values for the constants controlling the soft margin. For L_1 optimization the values used for the “-c” switch in SVMLight are $\{10^{-5}, 10^{-4}, 10^{-3}, \ldots, 10^{4}\}$. For the L_2 optimization, we used the following values for λ: $\{10^{-4}, 10^{-3}, \ldots, 10^{3}\}$. The results for SVM are given for the best parameters in a range of parameters tried. Thus these are parameter search values and essentially give upper bounds on the performance of SVM on these data sets. As can be seen the perceptron variants give similar accuracies and smaller variance and they are therefore an excellent alternative for SVM.

4. Discussion and Conclusions

The paper provides an experimental evaluation of several noise tolerant variants of the perceptron algorithm. The results are surprising since they suggest that the perceptron with margin is the most successful variant although it is the only one not designed for noise tolerance. The voted perceptron has similar performance in most cases, and it has the advantage that no parameter selection is required for it. The difference between voted and perceptron with margin are most noticeable in the artificial data sets, and the two are indistinguishable in their performance on the UCI data. The experiments also show that the soft-margin variants are generally weaker than voted or margin based algorithm and they do not provide additional improvement in performance when combined with these. Both the voted perceptron and the margin variant reduced the deviation in accuracy in addition to improving the accuracy. This is an important property that adds to the stability of the algorithms. Combining voted and perceptron with margin has the potential for further improvements but can harm performance in high variance cases. In terms of run time, the voted perceptron does not require parameter selection and can therefore be faster to train. On the other hand its test time is slower especially if one runs the primal version of the algorithm.

Overall, the results suggest that a good tradeoff is obtained by fixing a small value of τ; this gives significant improvements in performance without the training time penalty for parameter optimization or the penalty in test time for voting. In addition, since we have ruled out the use of the soft margin α variant we no longer need to run the algorithms for a large number of iterations; on the other hand, although on-line to batch conversion guarantees only hold for one iteration, experimental evidence suggests that a small number of iterations is typically helpful and sufficient.

Our results also highlight the problems involved with parameter selection. The method of double cross-validation is time intensive and our experiments for the large data sets were performed using the primal form of the algorithms since the dual form is too slow. In practice, with a large data set one can use a hold-out set for parameter selection so that run time is more manageable. In any case, such results must be accompanied by estimates of the deviation to provide a meaningful interpretation.

As mentioned in the introduction a number of variants that perform margin based updates exist in the literature (Friess et al., 1998; Gentile, 2001; Li and Long, 2002; Crammer et al., 2005; Kivinen et al., 2004; Shalev-Shwartz and Singer, 2005; Tsampouka and Shawe-Taylor, 2005). The algorithms vary in some other details. Aggressive ROMMA (Li and Long, 2002) explicitly maximizes the margin on the new example, relative to an approximation of the constraints from previous examples. NORMA (Kivinen et al., 2004) performs gradient descent on the soft margin risk resulting in an algorithm that rescales the old weight vector before the additive update. The Passive-Aggressive Algorithm (Crammer et al., 2005) adapts η on each example to guarantee it is immediately separable with margin (although update size is limited per noisy data). The Ballseptron (Shalev-Shwartz and Singer, 2005) establishes a normalized margin and replaces margin updates with updates on hypothetical examples on which a mistake would be made. ALMA (Gentile, 2001) renormalizes the weight vector so as to establish a normalized margin. ALMA is distinguished by tuning its parameters automatically during the on-line session, as well as having "p-norm variants" that can lead to other tradeoffs improving performance, for example, when the target is sparse. The algorithms of Tsampouka and Shawe-Taylor (2005) establish normalized margin by different normalization schemes. Following Freund and Schapire (1999) most of these algorithms come with mistake bound guarantees (or relative loss bounds) for the unrealizable case, that is, relative to the best hypothesis in a comparison class. Curiously, identical or similar bounds hold for the basic perceptron algorithm so that these results do not establish an advantage of the variants over the basic perceptron.

Another interesting algorithm, the Second Order Perceptron (Cesa-Bianchi et al., 2005), does not perform margin updates but uses spectral properties of the data in the updates in a way that can reduce mistakes in some cases.

Additional variants also exist for the on-line to batch conversions. Littlestone (1989) picks the best hypothesis using cross validation on a separate validation set. The Pocket algorithm with ratchet (Gallant, 1990) evaluates the hypotheses on the entire training set and picks the best, and the scheme of Cesa-Bianchi et al. (2004) evaluates each hypothesis on the remainder of the training set (after it made a mistake and is replaced) and adjusts for the different validation set sizes. The results in Cesa-Bianchi et al. (2004) show that loss bounds for the on line setting can be translated to error bounds in the batch setting even in the unrealizable case. Finally, experiments with aggressive ROMMA (Li and Long, 2002; Li, 2000) have shown that adding a voted perceptron scheme can harm performance, just as we observed for the margin perceptron. To avoid this, Li (2000) develops a scheme that appears to work well where the voting is done on a tail of the sequence of hypotheses which is chosen adaptively (see also Dekel and Singer, 2005, for more recent work).

Consider the noise tolerance guarantees for the unrealizable case. Ideally, one would want to find a hypothesis whose error rate is only a small additive factor away from the error rate of the best hypothesis in the class of hypotheses under consideration. This is captured by the agnostic PAC learning model (Kearns et al., 1994). It may be worth pointing out here that, although we have relative loss bounds for several variants and these can be translated to some error bounds in the batch setting, the bounds are not sufficiently strong to provide significant guarantees in the agnostic PAC model. So this remains a major open problem to be solved.

Seeing our experimental results in light of the discussion above there are several interesting questions. The first is whether the more sophisticated versions of margin perceptron add significant performance improvements. In particular it would be useful to know what normalization scheme is useful in what contexts so they can be clearly applied. We have raised parameter tuning as an important issue in terms of run time and the self tuning capacity of ALMA and related algorithms seems promising as an effective solution. Given the failure of the simple longest survivor it would be useful to evaluate the more robust versions of Gallant (1990) and Cesa-Bianchi et al. (2004). Notice, however that these methods have a cost in training time. Alternatively, one could further investigate the tail variants of the voting scheme (Li, 2000) or the "averaging" version of voting (Freund and Schapire, 1999; Gentile, 2001), explaining when they work with different variants and why. Finally, as mentioned above, to our knowledge there is no theoretical explanation to the fact that perceptron with margin performs better than the basic perceptron in the presence of noise. Resolving this is an important problem.

Acknowledgments

This work was partially supported by NSF grant IIS-0099446 and by Research Semester Fellowship Award from Tufts University. Many of the experiments were run on the Research Linux Cluster provided by Tufts Computing and Communications Services. We are grateful to Phil Long for pointing out the results in Li (2000) and the reviewers for pointing out Littlestone (1989) and Cesa-Bianchi et al. (2004) and for comments that helped improve the presentation of the paper.

References

B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 144–152, 1992.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *Information Theory, IEEE Transactions on*, 50(9):2050–2057, 2004.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. A second-order Perceptron algorithm. *SIAM Journal on Computing*, 34(3):640–668, 2005.

W. Cohen. Fast effective rule induction. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123, 1995.

M. Collins. Ranking algorithms for named entity extraction: Boosting and the voted perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 489–496, 2002.

K. Crammer, O. Dekel, S. Shalev-Shwartz, and Y. Singer. Online passive aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2005.

N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines*. Cambridge University Press, 2000.

O. Dekel and Y. Singer. Data driven online to batch conversions. In *Advances in Neural Information Processing Systems 18 (Proceedings of NIPS 2005)*, 2005.

T. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40:139–158, 2000.

T. Dietterich, M. Kearns, and Y. Mansour. Applying the weak learning framework to understand and improve c4.5. In *Proceedings of the International Conference on Machine Learning*, pages 96–104, 1996.

Y. Freund and R. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37:277–296, 1999.

T. Friess, N. Cristianini, and C. Campbell. The kernel adatron algorithm: A fast and simple learning procedure for support vector machines. In *Proceedings of the International Conference on Machine Learning*, pages 188–196, 1998.

S. Gallant. Perceptron-based learning algorithms. *IEEE Transactions on Neural Networks*, 1(2): 179–191, 1990.

A. Garg and D. Roth. Margin distribution and learning algorithms. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 210–217, 2003.

C. Gentile. A new approximate maximal margin classification algorithm. *Journal of Machine Learning Research*, 2:213–242, 2001.

T. Graepel, R. Herbrich, and R. Williamson. From margin to sparsity. In *Advances in Neural Information Processing Systems 13 (Proceedings of NIPS 2000)*, pages 210–216, 2000.

T. Joachims. Making large-scale svm learning practical. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1999.

M. Kearns, M. Li, L. Pitt, and L. G. Valiant. Recent results on boolean concept learning. In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 337–352, 1987.

M. Kearns, R. Schapire, and L. Sellie. Toward efficient agnostic learning. *Machine Learning*, 17 (2/3):115–141, 1994.

J. Kivinen, A. J. Smola, and R. C. Williamson. Online learning with kernels. *IEEE Transactions on Signal Processing*, 52(8):2165–2176, 2004.

A. Kowalczyk, A. Smola, and R. Williamson. Kernel machines and boolean functions. In *Advances in Neural Information Processing Systems 14 (Proceedings of NIPS 2001)*, pages 439–446, 2001.

W. Krauth and M. Mézard. Learning algorithms with optimal stability in neural networks. *Journal of Physics A*, 20(11):745–752, 1987.

Y. Li. Selective voting for perception-like online learning. In *Proceedings of the International Conference on Machine Learning*, pages 559–566, 2000.

Y. Li and P. Long. The relaxed online maximum margin algorithm. *Machine Learning*, 46:361–387, 2002.

Y. Li, H. Zaragoza, R. Herbrich, J. Shawe-Taylor, and J. Kandola. The perceptrom algorithm with uneven margins. In *International Conference on Machine Learning*, pages 379–386, 2002.

N. Littlestone. From on-line to batch learning. In *Proceedings of the Conference on Computational Learning Theory*, pages 269–284, 1989.

T.M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

D.J. Newman, S. Hettich, C.L. Blake, and C.J. Merz. UCI repository of machine learning databases, 1998. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

A. B. Novikoff. On convergence proofs on perceptrons. *Symposium on the Mathematical Theory of Automata*, 12:615–622, 1962.

F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.

R.A. Servedio. On PAC learning using Winnow, Perceptron, and a Perceptron-like algorithm. In *Proceedings of the Twelfth Annual Conference on Computational learning theory*, pages 296–307, 1999.

S. Shalev-Shwartz and Y. Singer. A new perspective on an old perceptron algorithm. In *Proceedings of the Sixteenth Annual Conference on Computational Learning Theory*, pages 264–278, 2005.

P. Tsampouka and J. Shawe-Taylor. Analysis of generic perceptron-like large margin classifiers. In *Proceedings of the European Conference on Machine Learning*, pages 750–758, 2005.

L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

Journal of Machine Learning Research 8 (2007) 249-276 Submitted 6/06; Revised 10/06; Published 2/07

Learnability of Gaussians with Flexible Variances

Yiming Ying Y.YING@CS.UCL.AC.UK
Department of Computer Science
University College London
Gower Street, London, WC1E 6BT, UK

Ding-Xuan Zhou MAZHOU@CITYU.EDU.HK
Department of Mathematics
City University of Hong Kong
Tat Chee Avenue, Kowloon, Hong Kong, China

Editor: Peter L. Bartlett

Abstract

Gaussian kernels with flexible variances provide a rich family of Mercer kernels for learning algorithms. We show that the union of the unit balls of reproducing kernel Hilbert spaces generated by Gaussian kernels with flexible variances is a uniform Glivenko-Cantelli (uGC) class. This result confirms a conjecture concerning learnability of Gaussian kernels and verifies the uniform convergence of many learning algorithms involving Gaussians with changing variances. Rademacher averages and empirical covering numbers are used to estimate sample errors of multi-kernel regularization schemes associated with general loss functions. It is then shown that the regularization error associated with the least square loss and the Gaussian kernels can be greatly improved when flexible variances are allowed. Finally, for regularization schemes generated by Gaussian kernels with flexible variances we present explicit learning rates for regression with least square loss and classification with hinge loss.

Keywords: Gaussian kernel, flexible variances, learning theory, Glivenko-Cantelli class, regularization scheme, empirical covering number

1. Introduction

Let X be a metric space in $\mathbb{R}^n$ and $Y \subseteq \mathbb{R}$. In learning theory, we are interested in the approximation of functions or function relations from samples. The functions are from an input space X to an output space Y, and samples are drawn according to a Borel probability measure ρ on the space $Z := X \times Y$. The target function f_ρ^V that we want to learn or approximate is a minimizer (may not be unique) of some error functional $\mathcal{E}(f) = \int_Z V(y, f(x))d\rho$ induced by a loss function $V : Y \times Y \to \mathbb{R}_+$, that is,

$$f_\rho^V = \arg\min \Big\{ \mathcal{E}(f) : \ f \text{ is a measurable function from } X \text{ to } Y \Big\}. \tag{1}$$

To define f_ρ^V, we denote ρ_X as the marginal distribution of ρ on X and $d\rho(\cdot|x)$ the conditional distribution. Then the error can be written as $\mathcal{E}(f) = \int_X \int_Y V(y, f(x))d\rho(y|x)d\rho_X(x)$, and we choose f_ρ^V to be a minimizer of the pointwise error: for almost every $x \in X$,

$$f_\rho^V(x) = \arg\min_{t \in Y} \int_Y V(y,t)d\rho(y|x).$$

The idea of many learning algorithms is to approximate f_ρ^V or its generalizing ability (measured by the error in terms of the loss V) by minimizing the empirical error $\mathcal{E}_\mathbf{z}(f) = \frac{1}{m}\sum_{i=1}^m V(y_i, f(x_i))$, or a penalized version, where $\mathbf{z} = \{(x_i, y_i)\}_{i=1}^m$ is a set of samples drawn independently according to ρ. The law of large numbers tells us that $\mathcal{E}_\mathbf{z}(f) \to \mathcal{E}(f)$ in probability for a fixed function f. This leads to the natural expectation that a minimizer $f_\mathbf{z}$ of $\mathcal{E}_\mathbf{z}$ over a set of functions $\mathcal{G}$, called the hypothesis space, would approximate a minimizer $f_\mathcal{G}$ of $\mathcal{E}$ in $\mathcal{G}$ (whose error is close to that of f_ρ^V when $\mathcal{G}$ is large): $\mathcal{E}(f_\mathbf{z}) \to \mathcal{E}(f_\mathcal{G})$ as $m \to \infty$. This approximation behaves well when the hypothesis space $\mathcal{G}$ enjoys the uniform convergence property (see Vapnik, 1998; Alon et al., 1997).

Throughout the paper we choose Y to be a subset of $\mathbb{R}$, the loss function V has an extended domain $V : Y \times \mathbb{R} \to \mathbb{R}_+$, and $f_\rho^V(x)$ is defined to be a minimizer of $\min_{t\in\mathbb{R}} \int_Y V(y,t)d\rho(y|x)$. In particular, for the binary classification problem (Devroye et al., 1997), we take $Y = \{1, -1\}$ and $V(y,t) = \phi(yt)$ with $\phi : \mathbb{R} \to \mathbb{R}_+$. For the hinge loss $\phi(yt) = \max\{1 - yt, 0\}$ used in the support vector machine for classification (Cortes and Vapnik, 1995), the target function $f_\rho^V(x) = f_c(x)$ is called the *Bayes rule* defined as $f_c(x) = 1$ for $\rho(y = 1|x) \geq \rho(y = -1|x)$, and $f_c(x) = -1$ otherwise. In this case, the uniform convergence can be characterized by the finiteness of the VC-dimension of $\mathcal{G}$ (see, e.g., Vapnik, 1998).

For the regression problem, we choose $Y = \mathbb{R}$ and $V(y,t) = \psi(y - t)$ with $\psi : \mathbb{R} \to \mathbb{R}_+$. In particular, for the least square loss $\psi(y-t) = (y-t)^2$, $f_\rho^V(x) = f_\rho(x) = \int_Y y d\rho(y|x)$ is the regression function (induced by conditional means). When Y is a closed interval on $\mathbb{R}$, the uniform convergence of real-valued function space $\mathcal{G}$ can be characterized by the property: for every $\varepsilon > 0$, there holds

$$\lim_{\ell\to+\infty} \sup_\mu \Pr\left\{ \sup_{m\geq\ell} \sup_{f\in\mathcal{G}} \left| \frac{1}{m}\sum_{i=1}^m f(x_i) - \int_X f(x)d\mu \right| > \varepsilon \right\} = 0. \tag{2}$$

Here Pr denotes the probability with respect to the samples $x_1, x_2, \ldots$, independently drawn according to a Borel probability distribution μ on X. The supremum is taken with respect to all such probability distributions. Following Dudley et al. (1991), we say that $\mathcal{G}$ is *uniform Glivenko-Cantelli* (uGC) if it satisfies the equality (2) for any $\varepsilon > 0$, which is equivalent to the finiteness of V_γ-dimension for any $\gamma > 0$ (Alon et al., 1997), see Section 4.

In this paper, we restrict our attention to the uniform convergence of kernel-based learning algorithms. A function $K : X \times X \to \mathbb{R}$ is called a *reproducing kernel* if it is symmetric and positive semidefinite, that is, for any finite set of distinct points $\{x_1, \cdots, x_\ell\} \subset \Omega$, the matrix $(K(x_i, x_j))_{i,j=1}^\ell$ is positive semidefinite. If moreover, K is continuous, then we call such a reproducing kernel a *Mercer kernel*. The *reproducing kernel Hilbert space* (RKHS) $\mathcal{H}_K$ associated with the kernel K is defined to be the completion of the linear span of the set of functions $\{K_x = K(x, \cdot) : x \in X\}$ with the reproducing property (Aronszajn, 1950)

$$f(x) = \langle f, K_x \rangle_K, \qquad \forall x \in X, f \in \mathcal{H}_K. \tag{3}$$

Learning algorithms considered here can be stated as minimization problems in $\mathcal{H}_K$. The reproducing property (3) makes the minimization over $\mathcal{H}_K$ be realized by an optimization procedure in $\mathbb{R}^m$. Consider the Tikhonov regularization scheme (Evgeniou et al., 2000) associated with the kernel K, loss V and a regularization parameter $\lambda > 0$ defined as

$$\widetilde{f}_{\mathbf{z},\lambda} = \arg\min_{f\in\mathcal{H}_K} \left\{ \frac{1}{m}\sum_{i=1}^m V(y_i, f(x_i)) + \lambda\|f\|_K^2 \right\}.$$

The Representer Theorem (Schölkopf et al., 2001; Wahba, 1990) tells us $\widetilde{f}_{\mathbf{z},\lambda} = \sum_{i=1}^{m} c_i K_{x_i}$ with $(c_i)_{i=1}^{m} \in \mathbb{R}^m$. When V is convex with respect to the second variable, this leads to a convex optimization problem in $\mathbb{R}^m$. In particular, for the least square loss or the hinge loss, it is a convex quadratic programming optimization problem which can handle large data settings. For discussions on the error rates of these schemes, see, for example, Vapnik (1998); Zhang (2004); Steinwart and Scovel (2005). This paper aims at Tikhonov regularization schemes associated with a set of kernels.

1.1 Multi-kernel Regularization Schemes

Multi-kernel regularization schemes have attracted attention recently due to applications in multi-task learning (Evgeniou and Pontil, 2004; Lanckriet et al., 2004), mixture density estimation (Li and Barron, 1999; Rakhlin et al., 2005), multi-kernel regularized classifiers (Chapelle et al., 2002; Cristianini et al., 1998; Wu et al., 2007), and many others. These schemes involve a set of Mercer kernels $\{K^\sigma\}_{\sigma\in\Sigma}$ with an index set Σ and take the following form with the regularization parameter $\lambda > 0$

$$f_{\mathbf{z},\lambda} := \arg\min_{\sigma\in\Sigma} \min_{f\in\mathcal{H}_{K^\sigma}} \left\{ \frac{1}{m}\sum_{i=1}^{m} V(y_i, f(x_i)) + \lambda \|f\|_{K^\sigma}^2 \right\}, \tag{4}$$

where the loss function $V(y,\cdot)$ is usually convex for any $y \in Y$.

Throughout the paper we assume the existence of a solution to the optimization problem (4). This assumption is satisfied when Σ is a compact metric space and $K^\sigma(x,y)$ is continuous with respect to $\sigma \in \Sigma$ for each fixed pair $(x,y) \in X \times X$. See Wu et al. (2007). When $\Sigma = (0,b]$ (a noncompact index set) with $0 < b < \infty$, each K^σ is a Gaussian kernel and V is the least-square loss, the existence will be verified in Appendix B under some conditions on the sample $\mathbf{z}$.

When we consider the convergence of $\mathcal{E}(f_{\mathbf{z},\lambda})$ to $\mathcal{E}(f_\rho^V)$, we need to find the hypothesis space for the uniform convergence here. Since $f_{\mathbf{z},\lambda}$ is the minimizer in (4), with a special choice $f = 0$ we see that whenever $f_{\mathbf{z},\lambda} \in \mathcal{H}_{K^\sigma}$, the quantity $\lambda\|f_{\mathbf{z},\lambda}\|_{K^\sigma}^2$ is bounded by

$$\mathcal{E}_{\mathbf{z}}(f_{\mathbf{z},\lambda}) + \lambda\|f_{\mathbf{z},\lambda}\|_{K^\sigma}^2 \le \mathcal{E}_{\mathbf{z}}(0) + 0 \le \frac{1}{m}\sum_{i=1}^{m} V(y_i, 0) \le \|V(y,0)\|_{L_\rho^\infty(Z)}.$$

When ρ satisfies $\|V(y,0)\|_{L_\rho^\infty(Z)} \le M < \infty$ (a strong assumption for regression problems, excluding Gaussian noise), we have $\|f_{\mathbf{z},\lambda}\|_{K^\sigma} \le \sqrt{M/\lambda}$ for all $\mathbf{z} \in Z^m$. This leads to the following hypothesis set.

Definition 1 *The normalized hypothesis set $\mathcal{H}$ associated with Mercer kernels $\{K^\sigma\}_{\sigma\in\Sigma}$ is defined as*

$$\mathcal{H} = \cup_{\sigma\in\Sigma}\{f \in \mathcal{H}_{K^\sigma} : \|f\|_{K^\sigma} \le 1\}. \tag{5}$$

The above analysis tells us that if $\|V(y,0)\|_{L_\rho^\infty(Z)} \le M$, then $f_{\mathbf{z},\lambda} \in \sqrt{\frac{M}{\lambda}}\,\mathcal{H} = \left\{\sqrt{\frac{M}{\lambda}} f : f \in \mathcal{H}\right\}$ for almost every $\mathbf{z} \in Z^m$. For the study of uniform convergence and error analysis on the multi-kernel scheme (4), a basic question is whether $\mathcal{H}$ is uGC. This is the main question investigated in this paper. The reproducing property of RKHS plays an essential role in our subsequent investigations.

The first purpose of this paper is to show that the uGC property of $\mathcal{H}$ is equivalent to that of a smaller set consisting of fundamental functions from the RKHS.

Theorem 2 *Let $\{K^\sigma\}_{\sigma\in\Sigma}$ be a set of Mercer kernels on X with*

$$\kappa := \sup_{\sigma\in\Sigma}\sup_{x\in X}\sqrt{K^\sigma(x,x)} < \infty. \tag{6}$$

Then the set $\mathcal{H}$ defined by (5) is uGC if and only if the normalized fundamental set

$$\mathcal{F} = \mathcal{F}_\Sigma = \{K_x^\sigma : \sigma \in \Sigma, x \in X\} \tag{7}$$

is uGC.

Theorem 2 will be proved in Section 3. The reproducing property (3) tells us that $\|K_x^\sigma\|_{K^\sigma} = \sqrt{K^\sigma(x,x)} \leq \kappa$ for any $x \in X$ and $\sigma \in \Sigma$. Therefore, the normalized condition (6) on the kernels $\{K_\sigma : \sigma \in \Sigma\}$ is essential for $\frac{1}{\kappa}\mathcal{F} \subseteq \mathcal{H}$ (a scaling). Since $\mathcal{F}$ contains much less functions than $\mathcal{H}$, checking the uGC property for $\mathcal{F}$ is potentially much simpler than that for $\mathcal{H}$. We shall use this idea to establish the learnability of Gaussian kernels with flexible variances.

1.2 uGC Property for Gaussians with Flexible Variances

The second purpose of this paper is to verify the learnability of Gaussian kernels with flexible variances stated in the form of the following uGC property. The theorem will be proved in Section 4.

Theorem 3 *Let $n \in \mathbb{N}$ and X be any subset of $\mathbb{R}^n$. Let*

$$K^\sigma(x,y) = \exp\left\{-\sum_{i=1}^n \frac{(x_i-y_i)^2}{\sigma_i^2}\right\} \quad \text{for } \sigma = (\sigma_1,\ldots,\sigma_n) \in (0,+\infty)^n. \tag{8}$$

Define $\mathcal{H}$ by (5) with $\Sigma = (0,+\infty)^n$. Then $\mathcal{H}$ is uGC.

Note that each kernel in (8) is, in a way, normalized: its $C(X)$ norm is 1, ensuring the kernel to be uniformly bounded by 1 and (6) to be satisfied with $\kappa = 1$.

When X is compact and the index set is restricted to be $[a,+\infty)^n$ with $a > 0$, we know from Theorem 3 in Zhou (2003) that for any $s \in \mathbb{N}$, the set $\mathcal{H}$ defined by (5) is included in a ball of $C^s(X)$ with a finite radius. Hence, the closure of $\mathcal{H}$ in $C(X)$ is compact. This belongs to the well-studied case (Cucker and Smale, 2001; Cucker and Zhou, 2007) that the closure of the hypothesis space $\mathcal{H}$ is compact, and thereby satisfies the uniform convergence condition.

When $a = 0$ and the index set becomes $(0,\infty)^n$, the closure is not compact any more. This observation led the second author to raise the following open problem in Zhou (2003): if we denote

$$K_\sigma(x,y) = \exp\left\{-\frac{|x-y|^2}{\sigma^2}\right\}, \qquad \text{with } \sigma > 0, \tag{9}$$

is the function set

$$\mathcal{H}_0 = \cup_{\sigma>0}\left\{ f(x) = \textstyle\sum_{i=1}^\ell c_i K_\sigma(x_i,x) : \ \sum_{i,j=1}^\ell c_i K_\sigma(x_i,x_j) c_j \leq 1, \ x_i \in [0,1]^n, \text{ and } \ell \in \mathbb{N} \right\} \tag{10}$$

involving the Gaussian kernels (9) with isotropic variances uGC? Using Theorem 3, we know that the answer to the question is positive.

Theorem 4 *Let $X = [0,1]^n$ and K_σ be given by (9). Define $\mathcal{H}_0$ by (10). Then $\mathcal{H}_0$ is uGC. It is contained in the unit ball of $C(X)$, but its closure in $C(X)$ is not compact.*

Proof The first statement follows from Theorem 3 and the fact that any subset of a uGC set is uGC.

Since each function f from $\mathcal{H}_0$ satisfies $\|f\|_{C(X)} \leq 1$, this set is contained in the unit ball of $C(X)$.

To see the last statement, we apply the Arzelá-Ascoli Theorem (see Yosida, 1980, P.85) which asserts that a subset of $C(X)$ has compact closure if and only if it is bounded and equicontinuous. Take $f_\sigma(x) = K_\sigma(x,0) = \exp\{-\frac{|x|^2}{\sigma^2}\} \in \mathcal{H}_0$ with $\sigma \in (0,\infty)$. For the neighborhood $[0,\varepsilon)^n$ of 0 with $\varepsilon > 0$, we see that

$$\sup_{\sigma\in(0,\infty)} \sup_{t\in[0,\varepsilon)^n} |f_\sigma(t) - f_\sigma(0)| = \sup_{\sigma\in(0,\infty)} \left|\exp\{-\frac{n\varepsilon^2}{\sigma^2}\} - 1\right| = 1 \not\to 0.$$

Therefore, the set of functions $\mathcal{H}_0$ is not equicontinuous. By the Arzelá-Ascoli Theorem, the closure of $\mathcal{H}_0$ is not compact. ■

The rest of the paper is organized as follows. In the next section we show the implications of the above main theorems to the error analysis for the regularization scheme (4). In particular, we present error rates of two examples involving Gaussian kernels with flexible variances. Sections 3 and 4 are distributed to prove Theorem 2 and 3 respectively. In Sections 5 and 6 we develop error bounds for the multi-kernel regularization scheme (4), especially for Gassian kernels with flexible variances. This is the last purpose of this paper. We postpone the derivation of error rates for regression with least square loss and classification with hinge loss to the end of this paper.

2. Applications to Error Analysis: Two Examples

The theorems in Section 1 give us qualitative results on the learnability of multi-kernel scheme (4). They can be deepened, which yields quantitative error rates for $\mathcal{E}(f_{\mathbf{z},\lambda}) - \mathcal{E}(f_\rho^V)$. In particular, we expect to derive explicit rates for multi-kernel regularized learning algorithms associated with Gaussian kernels with flexible variances.

To see how $\mathcal{E}(f_{\mathbf{z},\lambda})$ approximate $\mathcal{E}(f_\rho^V)$, assume $f_{\mathbf{z},\lambda} \in \mathcal{H}_{K^\sigma}$ for some $\sigma \in \Sigma$ and choose some $\sigma' \in \Sigma$ and $f_\lambda \in \mathcal{H}_{K^{\sigma'}}$ called a *regularizing function* (Smale and Zhou, 2004). Write

$$\begin{aligned}\mathcal{E}(f_{\mathbf{z},\lambda}) - \mathcal{E}(f_\rho^V) &= \left\{\{\mathcal{E}(f_{\mathbf{z},\lambda}) - \mathcal{E}_{\mathbf{z}}(f_{\mathbf{z},\lambda})\} + \{\mathcal{E}_{\mathbf{z}}(f_\lambda) - \mathcal{E}(f_\lambda)\}\right\} - \lambda\|f_{\mathbf{z},\lambda}\|^2_{K^\sigma} \\ &\quad + \left\{(\mathcal{E}_{\mathbf{z}}(f_{\mathbf{z},\lambda}) + \lambda\|f_{\mathbf{z},\lambda}\|^2_{K^\sigma}) - (\mathcal{E}_{\mathbf{z}}(f_\lambda) + \lambda\|f_\lambda\|^2_{K^{\sigma'}})\right\} \\ &\quad + \left\{\mathcal{E}(f_\lambda) - \mathcal{E}(f_\rho^V) + \lambda\|f_\lambda\|^2_{K^{\sigma'}}\right\}.\end{aligned}$$

The definition (4) tells us that the middle term above is at most 0. The first term $\Big\{\{\mathcal{E}(f_{\mathbf{z},\lambda}) - \mathcal{E}_{\mathbf{z}}(f_{\mathbf{z},\lambda})\} + \{\mathcal{E}_{\mathbf{z}}(f_\lambda) - \mathcal{E}(f_\lambda)\}\Big\}$ is called the *sample error*. Its second part $\mathcal{E}_{\mathbf{z}}(f_\lambda) - \mathcal{E}(f_\lambda) = \frac{1}{m}\sum_{i=1}^m \xi(z_i) - E(\xi)$ involving a single random variable $\xi = V(y, f(x))$ on Z can be easily estimated. The last term called the *regularization error* is independent of the samples (Niyogi and Girosi, 1996; Cucker and Smale, 2001; Smale and Zhou, 2003) and measures the approximation ability of the multi-kernel space $\cup_{\sigma\in\Sigma}\mathcal{H}_{K^\sigma}$.

Definition 5 *The regularization error associated with the regularizing function* $f_\lambda \in \mathcal{H}_{K^\sigma}$ *with* $\sigma \in \Sigma$ *is defined as*

$$\mathcal{D}(\lambda) = \mathcal{E}(f_\lambda) - \mathcal{E}(f_\rho^V) + \lambda \|f_\lambda\|^2_{K^\sigma}.$$

The regularization error of the system (4) is

$$\widetilde{\mathcal{D}}(\lambda) = \min_{\sigma\in\Sigma} \min_{f\in\mathcal{H}_{K^\sigma}} \{\mathcal{E}(f) - \mathcal{E}(f_\rho^V) + \lambda\|f\|^2_{K^\sigma}\}, \tag{11}$$

where f_λ *takes the special form*

$$f_\lambda = f_\lambda^V := \arg\min_{\sigma\in\Sigma} \min_{f\in\mathcal{H}_{K^\sigma}} \{\mathcal{E}(f) + \lambda\|f\|^2_{K^\sigma}\}. \tag{12}$$

Thus we have the *error decomposition*:

$$\mathcal{E}(f_{\mathbf{z},\lambda}) - \mathcal{E}(f_\rho^V) \le \left\{ \{\mathcal{E}(f_{\mathbf{z},\lambda}) - \mathcal{E}_{\mathbf{z}}(f_{\mathbf{z},\lambda})\} + \{\mathcal{E}_{\mathbf{z}}(f_\lambda) - \mathcal{E}(f_\lambda)\} \right\} + \mathcal{D}(\lambda). \tag{13}$$

What is left for the error analysis is the term $\mathcal{E}(f_{\mathbf{z},\lambda}) - \mathcal{E}_{\mathbf{z}}(f_{\mathbf{z},\lambda})$. It involves a set of random variables $\xi_{\mathbf{z}} = V(y, f_{\mathbf{z},\lambda}(x))$ with $\mathbf{z} \in Z^m$. That is also the motivation to study the uniform convergence of $\mathcal{H}$ (Vapnik, 1998).

By the error decomposition (13), the learning rate depends on trading off the sample error and the regularization error. The decay of regularization error $\widetilde{\mathcal{D}}(\lambda)$ relies on the regularity (smoothness) of the target function f_ρ^V for most commonly used loss functions which will be discussed in Section 6. Sample error estimates depend on the capacity of the hypothesis space $\sqrt{M/\lambda}\,\mathcal{H}$ which can be studied (see, e.g., Koltchinskii and Panchenko, 2002) by means of its covering numbers and Rademacher complexity (see Section 5). However, it is not easy to estimate the Rademacher complexity of $\sqrt{M/\lambda}\,\mathcal{H}$. In Section 5 we provide an alternative way to estimate the sample error by computing the Rademacher complexity of the fundamental set $\mathcal{F}$.

Let us give two examples both involving the Gaussian kernels (8) with flexible variances to illustrate the learning rates whose proofs will be given in Section 6.

The first example is regularized regression with the least square loss $V(y,t) = (y-t)^2$. Here $Y = \mathbb{R}$. Then the *multi-kernel least square regularized regression algorithm* (4) associated with Gaussian kernels (8) can be written as

$$f_{\mathbf{z},\lambda} = \arg\min_{\sigma\in\Sigma} \min_{f\in\mathcal{H}_{K^\sigma}} \left\{ \frac{1}{m}\sum_{i=1}^m (y_i - f(x_i))^2 + \lambda\|f\|^2_{K^\sigma} \right\}. \tag{14}$$

By the special feature of the least square loss, the distance between $f_{\mathbf{z},\lambda}$ and the target function f_ρ is often measured by the weighted L^2 metric in $L^2_{\rho_X}$ defined as $\|f\|_{L^2_{\rho_X}} = (\int_X |f(x)|^2 d\rho_X)^{1/2}$. When ρ_X is the Lebesgue measure, we denote the metric as $\|f\|_{L^2(X)}$ in Example 1. Also, denote $H^s(X)$ to be the Sobolev space (e.g., Stein, 1970) with index $s > 0$ on X.

For this multi-kernel algorithm, we have the following learning rates achieved by special choices of the regularization parameter $\lambda = \lambda(m)$.

Example 1 *Let $X \subseteq \mathbb{R}^n$ be a domain with Lipschitz boundary. Define $f_{\mathbf{z},\lambda}$ by (14) with the Gaussians (8). Assume $f_\rho \in H^s(X)$ for $s > 0$ and $|y| \leq M_0$ almost surely.*

(1) If $n/2 < s \leq n/2+2$ then for any $0 < \varepsilon < 2s-n$, we have

$$\mathbb{E}\Big[\|f_{\mathbf{z},\lambda} - f_\rho\|^2_{L^2_{\rho_X}(X)}\Big] = O\Big((\log m)^{\frac{1}{2}} m^{-\frac{2s-n-\varepsilon}{4(4s-n-2\varepsilon)}}\Big) \quad \text{by taking } \lambda = m^{-\frac{2s-\varepsilon}{4(4s-n-2\varepsilon)}}.$$

(2) If X is bounded, ρ_X is the Lebesgue measure, and $s \leq 2$ then by choosing $\lambda = m^{-\frac{2s+n}{4(4s+n)}}$, we get

$$\mathbb{E}\Big[\|f_{\mathbf{z},\lambda} - f_\rho\|^2_{L^2(X)}\Big] = O\Big((\log m)^{\frac{1}{2}} m^{-\frac{s}{2(4s+n)}}\Big).$$

In the above example we are considering the function approximation (De Vito et al., 2005; Smale and Zhou, 2005) on a domain of $\mathbb{R}^n$, so the learning rate is poor if the dimension n is large. However, in many situations, the input space X is a low-dimensional manifold embedded in the large-dimensional space $\mathbb{R}^n$. In such a situation, the learning rates may be greatly improved. This will not be discussed in this paper because the discussion involves the function approximation on Riemannian manifolds (see, e.g., Ye and Zhou, 2007), which is out of our scope here.

The second example is regularized classification with the hinge loss $V(y,t) = (1-yt)_+ := \max\{1-yt, 0\}$. Here $Y = \{1,-1\}$ and we are interested in functions $\mathcal{C} : X \to Y$ called binary classifiers which divide X into two classes. The target function is the Bayes rule f_c.

The *multi-kernel SVM regularized classification algorithm* (4) associated with Gaussian kernels (8) is defined to be a minimizer of the following optimization problem

$$f_{\mathbf{z},\lambda} = \arg\min_{\sigma\in\Sigma} \min_{f\in\mathcal{H}_{K^\sigma}} \left\{\frac{1}{m}\sum_{i=1}^{m}(1-y_i f(x_i))_+ + \lambda\|f\|^2_{K^\sigma}\right\}. \tag{15}$$

Then the sign function $\mathrm{sgn}(f_{\mathbf{z},\lambda})$ is used as a classifier where $\mathrm{sgn}(f)(x) = 1$ for $f(x) \geq 0$ and $\mathrm{sgn}(f)(x) = -1$ otherwise.

The prediction power of classifiers is measured by the misclassification error. The *misclassification error* for a classifier $\mathcal{C} : X \to Y$ is defined to be

$$\mathcal{R}(\mathcal{C}) := \Pr\{\mathcal{C}(x) \neq y\} = \int_X P(y \neq \mathcal{C}(x)|x) d\rho_X. \tag{16}$$

The Bayes rule is the classifier which minimizes the misclassification error.

The error analysis of classification algorithms often aims at understanding the approximating behaviors of the *excess misclassification error*

$$\mathcal{R}(\mathrm{sgn}(f_{\mathbf{z},\lambda})) - \mathcal{R}(f_c)$$

as the sample size m becomes large. Our learning rate for the hinge loss assumes a separable condition which was introduced by Chen et al. (2004) as follows.

Definition 6 *We say that ρ is separable by $\mathcal{H}_\Sigma$ if there is some $f_{sp} \in \mathcal{H}_{K^\sigma}$ with some $\sigma \in \Sigma$ such that $yf_{sp}(x) > 0$ almost surely. It has separation exponent $\theta \in (0,\infty]$ if we can choose f_{sp} and positive constants Δ, c_θ such that $\|f_{sp}\|_{K^\sigma} = 1$ and*

$$\rho_X\left\{x \in X : |f_{sp}(x)| < \Delta t\right\} \leq c_\theta t^\theta, \qquad \forall t > 0. \tag{17}$$

Observe that condition (17) with $\theta = \infty$ is equivalent to

$$\rho_X\{x \in X : |f_{\rm sp}(x)| < \gamma t\} = 0, \qquad \forall\, 0 < t < 1.$$

That is, $|f_{\rm sp}(x)| \geq \gamma$ almost everywhere. Thus, separable distributions with separation exponent $\theta = \infty$ are exactly strictly separable distributions. Recall (Shawe-Taylor et al., 1998; Vapnik, 1998) that ρ is said to be *strictly separable* with margin $\gamma > 0$ if ρ is separable together with the requirement $yf_{\rm sp}(x) \geq \gamma$ almost everywhere. The separation condition is different from the Tsybakov's noise condition (Tsybakov, 2004): the former involves a function $f_{\rm sp}$ from $\mathcal{H}_{K^\sigma}$ and describes the approximation of the RKHS (Chen et al., 2004), while the latter is about the distribution ρ only.

The learning rates for the multi-kernel SVM algorithm (15) can be stated as follows.

Example 2 *Let $f_{\mathbf{z},\lambda}$ be defined by (15) with $\{K^\sigma\}$ given by (8). If ρ is separable by $\mathcal{H}_\Sigma$ with some separation exponent $\theta > 0$, then by choosing $\lambda = m^{-\frac{2+\theta}{2(2+3\theta)}}$, we have*

$$\mathbb{E}\Big[\mathcal{R}(sgn(f_{\mathbf{z},\lambda})) - \mathcal{R}(f_c)\Big] = O\Big((\log m)^{\frac{1}{2}} m^{-\frac{\theta}{2(2+3\theta)}}\Big).$$

It is observed in applications that allowing flexible variances improves the learnability of Gaussian kernels. As we shall see in Section 6 for the least square loss, when the regression function f_ρ has Sobolev smoothness, the regularization error (11) associated with Gaussians with flexible variances decays as $O(\lambda^s)$ for some $s > 0$. This has also been confirmed theoretically by Steinwart and Scovel (2005) for classification with the hinge loss, under some geometric noise condition for the distribution. Such a decay is impossible for the regularization error associated with a single Gaussian kernel, at least when ρ_X is the Lebesgue measure on X, as shown by Smale and Zhou (2003). This demonstrates that learning algorithms using Gaussian kernels with flexible variances have advantages for many applications.

Another way to obtain improved error rates is to take kernels changing with the sample size. Kernels of this kind include polynomial kernels with changing degrees (Zhou and Jetter, 2006) and Gaussian kernels with changing variances (Steinwart, 2001; Steinwart and Scovel, 2005). Though the learning rates given by Steinwart and Scovel (2005) is comparable to those in Example 2, the main difficulty there is a requirement of some information about the distribution ρ for the choice of the kernel (similar to the choice of the regularization parameter according to some regularity properties of ρ). Compared to that, when no information about ρ is available, the algorithm (4) still produces the empirically optimal kernel from the kernel set, as part of the optimization problem.

3. Reducing the Hypothesis Set

In this section we show how the uGC property of the normalized hypothesis set can be reduced to that of the normalized fundamental set, hence establish Theorem 2. For $\sigma \in \Sigma$, denote $\mathcal{H}_\sigma = \mathcal{H}_{K^\sigma}, \langle\cdot,\cdot\rangle_\sigma = \langle\cdot,\cdot\rangle_{K^\sigma}$, and $\kappa^\sigma = \sup_{x\in X}\sqrt{K^\sigma(x,x)}$. Assume $\kappa^\sigma < \infty$.

Let μ be a Borel probability distribution on X. For $\{x_i\}_{i=1}^m$ drawn according to μ, we denote $\mathbb{E}_m(f) = \frac{1}{m}\sum_{i=1}^m f(x_i)$ and $\mathbb{E}(f) = \int_X f(x)d\mu$.

To prove Theorem 1, we need the following proposition about the function

$$F^\sigma(x) := \int_X K^\sigma(x,y)d\mu(y), \qquad x \in X. \tag{18}$$

Proposition 7 *Let K^σ be a Mercer kernel on X and μ be a Borel probability distribution. Define the function F^σ by (18). Then we have*

(a) $F^\sigma \in \mathcal{H}_\sigma$;

(b) $\langle f, F^\sigma \rangle_\sigma = \int_X f(y) d\mu(y)$ for every $f \in \mathcal{H}_\sigma$.

Proof Define a linear functional T_σ on $\mathcal{H}_\sigma$ as

$$T_\sigma(f) = \int_X f(y) d\mu(y).$$

Since μ is a Borel probability measure on X, we know by the reproducing property (3)

$$|T_\sigma(f)| \le \int_X |\langle f, K_y^\sigma \rangle_\sigma| d\mu(y) \le \kappa^\sigma \|f\|_\sigma, \qquad \forall f \in \mathcal{H}_\sigma.$$

This means T_σ is a bounded linear functional on the Hilbert space $\mathcal{H}_\sigma$. By the Riesz Representation Theorem of Hilbert spaces, we know that there exists a function $g^\sigma \in \mathcal{H}_\sigma$ such that

$$T_\sigma(f) = \langle f, g^\sigma \rangle_\sigma, \qquad \forall f \in \mathcal{H}_\sigma. \tag{19}$$

In particular, for the function $f = K_x^\sigma$ lying in $\mathcal{H}_\sigma$ with x being an arbitrarily fixed point in X, there holds

$$T_\sigma(K_x^\sigma) = \int_X K_x^\sigma(y) d\mu(y) = \int_X K^\sigma(x,y) d\mu(y) = \langle K_x^\sigma, g^\sigma \rangle_\sigma = g^\sigma(x).$$

This equals to $F^\sigma(x)$ according to the definition (18). Hence $F^\sigma(x) = g^\sigma(x)$ for every $x \in X$. Therefore $F^\sigma = g^\sigma \in \mathcal{H}_\sigma$ which proves property (a). Property (b) is an immediate consequence of equality (19) since $g^\sigma = F^\sigma$. ■

Property (a) of Proposition 7 means that the integral and the inner product with K_y^σ can be interchanged:

$$\langle f, \int_X K^\sigma(\cdot, y) d\mu(y) \rangle_\sigma = \int_X f(y) d\mu(y) = \int_X \langle f, K_y^\sigma \rangle_\sigma d\mu(y).$$

Lemma 8 *Let K^σ be a Mercer kernel on X and μ be a Borel probability distribution. Denote $G^\sigma(x) = \frac{1}{m}\sum_{i=1}^m K^\sigma(x_i, x) - F^\sigma(x)$ for $m \in \mathbb{N}$ and $\mathbf{x} = (x_i)_{i=1}^m \in X^m$. Then, the following statements are true.*

(a) $G^\sigma \in \mathcal{H}_\sigma$ and $\sup_{f \in \mathcal{H}_\sigma, \|f\|_\sigma \le 1} |\mathbb{E}_m(f) - \mathbb{E}(f)| = \|G^\sigma\|_\sigma$.

(b) $\|G^\sigma\|_\sigma \le \sqrt{2\|G^\sigma\|_{C(X)}}$ and $\|G^\sigma\|_{C(X)} \le \kappa^\sigma \|G^\sigma\|_\sigma$.

Proof By property (a) of Proposition 7, $G^\sigma = \frac{1}{m}\sum_{i=1}^m K_{x_i}^\sigma - F^\sigma \in \mathcal{H}_\sigma$. This in connection with (3) and property (b) of Proposition 7 tells us that for any $f \in \mathcal{H}_\sigma$,

$$\mathbb{E}_m(f) - \mathbb{E}(f) = \langle f, \frac{1}{m}\sum_{i=1}^m K_{x_i}^\sigma - F^\sigma \rangle_\sigma = \langle f, G^\sigma \rangle_\sigma.$$

Then

$$\sup_{f \in \mathcal{H}_\sigma, \|f\|_\sigma \le 1} |\mathbb{E}_m(f) - \mathbb{E}(f)| = \sup_{f \in \mathcal{H}_\sigma, \|f\|_\sigma \le 1} |\langle f, G^\sigma \rangle_\sigma| = \|G^\sigma\|_\sigma.$$

This proves the first statement.

To verify the second statement, we compute the norm $\|G^\sigma\|_\sigma$ by the definition and property (b) of Proposition 7. It yields

$$\|G^\sigma\|_\sigma^2 = \frac{1}{m^2}\sum_{i,j=1}^m K^\sigma(x_i,x_j) - \frac{2}{m}\sum_{i=1}^m \int_X K^\sigma_{x_i}(y)d\mu(y) + \int_X F^\sigma(y)d\mu(y).$$

But $\int_X K^\sigma_{x_i}(y)d\mu(y) = \int_X K^\sigma(x_i,y)d\mu(y) = F^\sigma(x_i)$. So we have

$$\begin{aligned}\|G^\sigma\|_\sigma^2 &= \tfrac{1}{m}\sum_{i=1}^m \left\{\frac{1}{m}\sum_{j=1}^m K^\sigma(x_i,x_j) - F^\sigma(x_i)\right\} \\ &\quad - \int_X \left\{\frac{1}{m}\sum_{i=1}^m K^\sigma(x_i,y) - F^\sigma(y)\right\}d\mu(y) \\ &= \frac{1}{m}\sum_{i=1}^m G^\sigma(x_i) - \int_X G^\sigma(y)d\mu(y) \\ &\le 2\sup_{y\in X}|G^\sigma(y)| = 2\|G^\sigma\|_{C(X)}.\end{aligned}$$

This gives the first inequality of the second statement.

The second inequality of the second statement follows directly from the reproducing property:

$$|G^\sigma(y)| = |\langle G^\sigma, K^\sigma_y\rangle_\sigma| \le \|G^\sigma\|_\sigma \|K^\sigma_y\|_\sigma \le \kappa^\sigma \|G^\sigma\|_\sigma, \qquad \forall y \in X.$$

This completes the proof of the lemma. ■

Now we are ready to prove Theorem 2 stated in Section 1.

Proof of Theorem 2. Recall $\kappa = \sup_{\sigma\in\Sigma}\kappa^\sigma$ and the definition (5) of the set $\mathcal{H}$. By Part (a) of Lemma 8,

$$\sup_{f\in\mathcal{H}}|\mathbb{E}_m(f) - \mathbb{E}(f)| = \sup_{\sigma\in\Sigma}\sup_{f\in\mathcal{H}_\sigma, \|f\|_\sigma\le 1}|\mathbb{E}_m(f) - \mathbb{E}(f)| = \sup_{\sigma\in\Sigma}\|G^\sigma\|_\sigma.$$

On the other hand, since $\{K^\sigma_y : \sigma\in\Sigma, y\in X\}$ is exactly the set $\mathcal{F}$ according to its definition (7), we see that

$$\sup_{f\in\mathcal{F}}|\mathbb{E}_m(f) - \mathbb{E}(f)| = \sup_{\sigma\in\Sigma}\sup_{y\in X}\left|\frac{1}{m}\sum_{i=1}^m K^\sigma_y(x_i) - F^\sigma(y)\right| = \sup_{\sigma\in\Sigma}\|G^\sigma\|_{C(X)}.$$

This in connection with Lemma 8 implies that for any $\varepsilon > 0$ and $\ell \in \mathbb{N}$, there holds

$$\begin{aligned}\left\{\sup_{m\ge\ell}\sup_{f\in\mathcal{F}}|\mathbb{E}_m(f) - \mathbb{E}(f)| > \kappa\varepsilon\right\} &\subseteq \left\{\sup_{m\ge\ell}\sup_{f\in\mathcal{H}}|\mathbb{E}_m(f) - \mathbb{E}(f)| > \varepsilon\right\} \\ &\subseteq \left\{\sup_{m\ge\ell}\sup_{f\in\mathcal{F}}|\mathbb{E}_m(f) - \mathbb{E}(f)| > \tfrac{\varepsilon^2}{2}\right\}.\end{aligned}$$

Therefore, $\mathcal{H}$ is uGC if and only if $\mathcal{F}$ is. This proves Theorem 2. ■

4. Gaussian Kernels Provide uGC Hypothesis Sets

In this section we establish the uGC property of Gaussians with flexible variances stated in Theorem 3. This will be done by verifying a criterion for uGC sets in terms of empirical covering numbers given by Dudley et al. (1991). For $1 \leq p < +\infty$ and $\mathbf{x} = (x_i)_{i=1}^m \in X^m$, we denote the l^p empirical metric of two functions f, g as

$$d_{\mathbf{x},p}(f,g) = \Big(\frac{1}{m}\sum_{i=1}^m |f(x_i) - g(x_i)|^p\Big)^{1/p}$$

and for $p = \infty$

$$d_{\mathbf{x},\infty}(f,g) = \max_{1\leq i\leq m} |f(x_i) - g(x_i)|.$$

Definition 9 *Let $\mathcal{G}$ be a set of functions on X, $1 \leq p \leq +\infty$ and $\mathbf{x} = (x_i)_{i=1}^m \in X^m$. The empirical covering number $\mathcal{N}_p(\mathcal{G}, \mathbf{x}, \eta)$ is defined to be the minimal integer N such that there are N functions $\{g^j\}_{j=1}^N \subset \mathcal{G}$ satisfying*

$$\min_{1\leq j\leq N} d_{\mathbf{x},p}(g, g^j) \leq \eta, \qquad \forall g \in \mathcal{G}.$$

The metric entropy of $\mathcal{G}$ is defined as

$$H_{m,p}(\mathcal{G}, \eta) = \sup_{\mathbf{x}\in X^m} \log \mathcal{N}_p(\mathcal{G}, \mathbf{x}, \eta), \qquad m \in \mathbb{N}, \eta > 0.$$

The following criterion was established by Dudley et al. (1991).

Lemma 10 *A set $\mathcal{G}$ of functions from X to $[0,1]$ is uGC if and only if for some $1 \leq p \leq +\infty$, there holds*

$$\lim_{m\to\infty} \frac{H_{m,p}(\mathcal{G}, \eta)}{m} = 0, \qquad \forall \eta > 0.$$

To continue, we need the following combinatorial dimensions.

Definition 11 *Let $\mathcal{G}$ be a set of functions from X to $[0,1]$. We say that $A \subset X$ is V_γ shattered (P_γ shattered) by $\mathcal{G}$ if there is a number $\alpha \in \mathbb{R}$ (a function $s : A \to [0,1]$) with the following property: For every subset E of A there exists some function $f_E \in \mathcal{G}$ such that $f_E(x) \leq \alpha - \gamma$ ($f_E(x) \leq s(x) - \gamma$) for every $x \in A \setminus E$, and $f_E(x) \geq \alpha + \gamma$ ($f_E(x) \geq s(x) + \gamma$) for every $x \in E$. The V_γ dimension of $\mathcal{G}$, $V_\gamma(\mathcal{G})$, (The P_γ dimension of $\mathcal{G}$, $P_\gamma(\mathcal{G})$,) is the maximal cardinality of a set $A \subset X$ that is V_γ shattered (P_γ shattered) by $\mathcal{G}$.*

Based on Lemma 10, Alon et al. (1997) showed that $\mathcal{G}$ is uGC if and only if the V_γ dimension or P_γ dimension of $\mathcal{G}$ is finite for every $\gamma > 0$.

In addition, we need the following relation between the dimensions and the empirical covering numbers essentially proved by Alon et al. (1997). A complete proof is given in Appendix A.

Lemma 12 *Let $\mathcal{G}$ be a set of functions from X to $[0,1]$.*

(a) $V_\gamma(\mathcal{G}) \leq P_\gamma(\mathcal{G}) \leq (\frac{2}{\gamma}+1)V_{\gamma/2}(\mathcal{G})$ for any $\gamma > 0$.

(b) For every $m \in \mathbb{N}$ and $0 < \varepsilon < 1$, there holds

$$\sup_{\mathbf{x}\in X^m} \mathcal{N}_\infty(\mathcal{G},\mathbf{x},\varepsilon) \le 2\left(\frac{4m}{\varepsilon^2}\right)^{1+d\log\left(\frac{2em}{\varepsilon}\right)}, \qquad d := P_{\varepsilon/4}(\mathcal{G}).$$

With the above preparations, we can prove Theorem 3. First, let us begin with the univariate case.

Lemma 13 *Let X be a subset of $\mathbb{R}$ and $\mathcal{F}$ be given by (7) with $K^\sigma(x,y) = \exp\{-\frac{(x-y)^2}{\sigma^2}\}$ and $\Sigma = (0,+\infty)$. Then $V_\gamma(\mathcal{F}) \le 2$ for every $\gamma > 0$.*

Proof Suppose to the contrary that $V_\gamma(\mathcal{F}) \ge 3$ for some $\gamma > 0$. It means that there is a set $A = \{x_1,x_2,x_3\} \subset X$ with $x_1 < x_2 < x_3$ which is V_γ-shattered by $\mathcal{F}$. That is, there is some $\alpha \in \mathbb{R}$ such that for every $E \subseteq A$ there exists some function $f_E \in \mathcal{F}$ satisfying

$$\begin{cases} f_E(x) \le \alpha - \gamma, & \text{for } x \in A\setminus E, \\ f_E(x) \ge \alpha+\gamma, & \text{for } x \in E. \end{cases}$$

Take $E = \{x_1,x_3\} \subset A$. Then there is a function $f_E \in \mathcal{F}$ such that $f_E(x_2) \le \alpha-\gamma$ and $f_E(x_1)$, $f_E(x_3) \ge \alpha+\gamma$. The function f_E can be represented as $f_E(x) = \exp\{-\frac{(x-y)^2}{\sigma^2}\}$ for some $y \in X$ and $\sigma \in (0,\infty)$. It can be extended to the whole real line with the same expression and we denote this extended function as $\widetilde{f_E}$. The function $\widetilde{f_E}$ has no local minimum on $\mathbb{R}$. However, it is continuous and satisfies

$$\widetilde{f_E}(x_1) = f_E(x_1) > f_E(x_2) = \widetilde{f_E}(x_2), \qquad \widetilde{f_E}(x_3) = f_E(x_3) > f_E(x_2) = \widetilde{f_E}(x_2).$$

So the minimum value of $\widetilde{f_E}$ on the closed interval $[x_1,x_3]$ is achieved on the open interval (x_1,x_3). Consequently, $\widetilde{f_E}$ has a local minimum on $\mathbb{R}$, which is a contradiction. ■

Next, we prove Theorem 3 by Lemmas 10 and 13. Here the tensor product form of the functions in the set $\mathcal{F}$ plays an essential role.

Proof of Theorem 3. By Theorem 2, we need to show that $\mathcal{F}$ is uGC. Consider another set of functions on $\mathbb{R}^n$ defined as

$$\widetilde{\mathcal{F}} = \{K_x^\sigma = K^\sigma(x,\cdot) : \sigma \in \Sigma, x \in \mathbb{R}^n\}.$$

If $\widetilde{\mathcal{F}}$ is uGC as a set of functions from $\mathbb{R}^n$ to $[0,1]$, then its restriction onto X is also uGC, which would imply the uGC property of $\mathcal{F}$. Therefore, it suffices to prove the uGC of $\widetilde{\mathcal{F}}$.

Define sets of univariate functions as

$$\mathcal{F}^j = \left\{e^{-(t-s_j)^2/\sigma_j^2} : \sigma_j \in (0,\infty), s_j \in \mathbb{R}\right\}, \qquad j = 1,2,\dots,n.$$

Fix $j \in \{1,2,\dots,n\}$. Consider the set $\mathcal{F}^j$ of functions from $\mathbb{R}$ to $[0,1]$. Lemma 13 tells us that $V_\gamma(\mathcal{F}^j) \le 2$ for every $\gamma > 0$. Applying Lemma 12 to $\mathcal{G} = \mathcal{F}^j$ of functions on $\mathbb{R}$, we find that for every $0 < \varepsilon < 1$, $P_{\varepsilon/4}(\mathcal{F}^j) \le \frac{16}{\varepsilon} + 2$ and

$$\sup_{\mathbf{t}\in\mathbb{R}^m} \mathcal{N}_\infty(\mathcal{F}^j,\mathbf{t},\varepsilon) \le \mathcal{N}_0 := 2\left(\frac{4m}{\varepsilon^2}\right)^{1+\left(\frac{16}{\varepsilon}+2\right)\log\left(\frac{2em}{\varepsilon}\right)}. \tag{20}$$

Now we apply (20) to estimate the empirical covering number of the set $\widetilde{\mathcal{F}}$. To this end, let $0<\varepsilon<1$ and $\mathbf{x}=(x_i)_{i=1}^m$ where each point $x_i\in\mathbb{R}^n$ can be expressed as a vector $x_i=(x_{i,1},\ldots,x_{i,n})$. Let $j\in\{1,2,\ldots,n\}$. Consider $\mathbf{t}:=(x_{i,j})_{i=1}^m\in\mathbb{R}^m$. The bound (20) tells us that there are $\mathcal{N}_0$ pairs $\{(\sigma_{j,\ell},s_{j,\ell})\}_{\ell=1}^{\mathcal{N}_0}$ with $\sigma_{j,\ell}\in(0,\infty)$ and $s_{j,\ell}\in\mathbb{R}$ representing $\mathcal{N}_0$ functions $\left\{g^\ell(t)=e^{-(t-s_{j,\ell})^2/\sigma_{j,\ell}^2}\right\}_{\ell=1}^{\mathcal{N}_0}$ $\subset\mathcal{F}^j$ such that

$$\min_{1\le\ell\le\mathcal{N}_0}d_{\mathbf{t}}(g,g^\ell)=\min_{1\le\ell\le\mathcal{N}_0}\left\{\max_{1\le i\le m}|g(x_{i,j})-g^\ell(x_{i,j})|\right\}\le\varepsilon,\qquad\forall g\in\mathcal{F}^j.$$

That is, for any $\sigma_j\in(0,\infty)$ and $s_j\in\mathbb{R}$, we can find some $\ell\in\{1,\ldots,\mathcal{N}_0\}$ satisfying

$$\left|\exp\left\{-\frac{(x_{i,j}-s_j)^2}{\sigma_j^2}\right\}-\exp\left\{-\frac{(x_{i,j}-s_{j,\ell})^2}{\sigma_{j,\ell}^2}\right\}\right|\le\varepsilon,\qquad\forall i=1,\ldots,m. \tag{21}$$

Now we choose a set $\mathcal{F}_\varepsilon$ of functions on $\mathbb{R}^n$ consisting of $\mathcal{N}_0^n$ functions

$$f_{\ell_1,\ell_2,\ldots,\ell_n}(\cdot)=\exp\left\{-\sum_{j=1}^n\frac{(\cdot_j-s_{j,\ell_j})^2}{\sigma_{j,\ell_j}^2}\right\},\qquad\ell_1,\ell_2,\ldots,\ell_n\in\{1,2,\ldots,\mathcal{N}_0\}.$$

Each function $f\in\widetilde{\mathcal{F}}$ can be expressed as

$$f(x_i)=\exp\left\{-\sum_{j=1}^n\frac{(x_{i,j}-s_j)^2}{\sigma_j^2}\right\}$$

with $\{\sigma_j\}_{j=1}^n\subset(0,\infty)^n$ and $\{s_j\}_{j=1}^n\subset\mathbb{R}^n$. We can choose some $\{\ell_j\}_{\ell=1}^n\in\{1,\ldots,\mathcal{N}_0\}^n$ satisfying (21). Then

$$|f(x_i)-f_{\ell_1,\ell_2,\ldots,\ell_n}(x_i)|\le\sum_{p=1}^n\exp\left\{-\sum_{j=p+1}^n\frac{(x_{i,j}-s_j)^2}{\sigma_j^2}\right\}\varepsilon\le n\varepsilon.$$

Thus, we have

$$d_{\mathbf{x},\infty}(f,f_{\ell_1,\ell_2,\ldots,\ell_n})=\max_{1\le i\le m}|f(x_i)-f_{\ell_1,\ell_2,\ldots,\ell_n}(x_i)|\le n\varepsilon.$$

By the definition of empirical covering numbers, we have

$$\mathcal{N}_\infty(\widetilde{\mathcal{F}},\mathbf{x},n\varepsilon)\le\mathcal{N}_0^n=2^n\left(\frac{4m}{\varepsilon^2}\right)^{n+\left(\frac{16}{\varepsilon}+2\right)n\log\left(\frac{2em}{\varepsilon}\right)}.$$

Therefore, for any $0<\varepsilon<1$,

$$H_{m,\infty}(\widetilde{\mathcal{F}},n\varepsilon)\le n\log 2+\left(n+\left(\frac{16}{\varepsilon}+2\right)n\log\left(\frac{2em}{\varepsilon}\right)\right)\log\frac{4m}{\varepsilon^2}. \tag{22}$$

Observe that $\mathcal{N}_\infty(\widetilde{\mathcal{F}},\mathbf{x},\varepsilon)=1$ for any $\varepsilon\ge1$. Hence $H_{m,\infty}(\widetilde{\mathcal{F}},n\varepsilon)=0$ for any $\varepsilon\ge1$. Combining this observation with (22) implies that

$$\lim_{m\to\infty}\frac{H_{m,\infty}(\widetilde{\mathcal{F}},n\varepsilon)}{m}=0,\qquad\text{for any }\varepsilon>0.$$

Hence $\widetilde{\mathcal{F}}$ is uGC by Lemma 10. This completes the proof of Theorem 3. ■

The proof of Theorem 3 actually gives estimates for the empirical covering number which can be used to bound the sample error in (13) as we shall do in the following section.

5. Error Bound by Rademacher Averages

In order to bound the error $\mathcal{E}(f_{\mathbf{z},\sigma}) - \mathcal{E}(f_\rho^V)$, we know from the error decomposition (13) that it is sufficient to estimate the sample error and the regularization error. In particular, we will provide the error bounds for the multi-kernel scheme (4) generated by Gaussians with flexible variances.

5.1 Sample Error Estimate

In this subsection we are mainly concerned about the sample error. The regularization error will be discussed in the next subsection. The estimate of the sample error for the regularized multi-kernel scheme (4) involves the hypothesis space

$$\mathcal{H}_\lambda = \bigcup_{\sigma\in\Sigma} \Big\{ f \in \mathcal{H}_{K_\sigma} : \|f\|_{K_\sigma} \le \sqrt{\frac{M}{\lambda}} \Big\}. \tag{23}$$

Below, we show how to get sample error estimates by Rademacher complexities (Bartlett and Mendelson, 2002; Koltchinskii, 2001; Koltchinskii and Panchenko, 2002) of the reduced hypothesis space $\mathcal{F}$ defined by (7), which is potentially easier to compute than that of $\mathcal{H}_\lambda$. Let's first introduce Rademacher average over a set of functions F on Ω.

Definition 14 *Let μ be a probability measure on Ω and F be a class of uniformly bounded functions. For every integer m, let*

$$R_m(F) := \mathbb{E}_\mu \mathbb{E}_\varepsilon \Big[\frac{1}{m} \sup_{f\in F} \Big| \sum_{i=1}^m \varepsilon_i f(z_i) \Big| \Big]$$

where $\{z_i\}_{i=1}^m$ are independent random variables distributed according to μ and $\{\varepsilon_i\}_{i=1}^m$ are independent Rademacher random variables, that is, $P(\varepsilon_i = +1) = P(\varepsilon_i = -1) = 1/2$.

Turn to the multi-kernel regularization scheme (4). If the loss function $V(y,t)$ is convex with respect to t, its left and right partial derivatives with respect to the second variable, denoted as $V'_-(y,t), V'_+(y,t)$ respectively for simplicity, exist for every $y \in Y$. Throughout this section, we assume the loss function is admissible in the following sense.

Definition 15 *We say that the loss function $V : Y \times \mathbb{R} \to \mathbb{R}_+$ is admissible (with respect to ρ) if it is convex with respect to the second variable, $M = \|V(y,0)\|_{L^\infty_\rho(Z)} < +\infty$ and, for any $\lambda > 0$ there holds*

$$C_\lambda = \sup\Big\{ \max(|V'_-(y,t)|, |V'_+(y,t)|) : y \in Y,\ |t| \le \kappa\sqrt{\frac{M}{\lambda}} \Big\} < +\infty. \tag{24}$$

We also need the following lemma (Bartlett and Mendelson, 2002; Ledoux and Talagrand, 1991) summarizing some of the properties of Rademacher averages. A complete proof is given in Appendix A.

Lemma 16 *Let F be a class of uniformly bounded real-valued functions on (Ω, μ) and $m \in \mathbb{N}$.*

(a) For every $c \in \mathbb{R}$, $R_m(cF) = |c| R_m(F)$, where $cF = \{cf : f \in F\}$.

(b) If for each $i \in \{1,\ldots,m\}$, $\phi_i : \mathbb{R} \to \mathbb{R}$ is a function with $\phi_i(0) = 0$ having a Lipschitz constant c_i, then for any $\{x_i\}_{i=1}^m$,

$$\mathbb{E}_\varepsilon\Big[\sup_{f\in F}\Big|\sum_{i=1}^m \varepsilon_i\phi_i(f(x_i))\Big|\Big] \le 2\mathbb{E}_\varepsilon\Big[\sup_{f\in F}\Big|\sum_{i=1}^m c_i\varepsilon_i f(x_i)\Big|\Big].$$

The sample error analysis of multi-kernel regularization scheme (4) involves the Rademacher complexity of the fundamental space $\mathcal{F}$ denoted by

$$R_m(\mathcal{F}) = \mathbb{E}_{\rho_X}\mathbb{E}_\varepsilon\Big[\frac{1}{m}\sup_{f\in\mathcal{F}}\Big|\sum_{i=1}^m \varepsilon_i f(x_i)\Big|\Big].$$

Lemma 17 *Let $\mathcal{H}_\lambda$ be defined by (23), then*

$$\mathbb{E}\Big[\sup_{f\in\mathcal{H}_\lambda}|\mathcal{E}(f)-\mathcal{E}_{\mathbf{z}}(f)|\Big] \le 4C_\lambda\sqrt{\frac{M}{\lambda}}\,\Big(R_m(\mathcal{F})\Big)^{1/2} + \frac{2M}{\sqrt{m}}.$$

Proof Let $\mathcal{V}_\lambda$ be a set of functions on Z defined as

$$\mathcal{V}_\lambda = \Big\{V(y,f(x)) :\ f\in\mathcal{H}_\lambda\Big\}.$$

Using standard symmetrization arguments (e.g., Van der Vaart and Weller, 1996, Lemma 2.3.1), one can see that

$$\mathbb{E}_\rho\mathbb{E}_\varepsilon\Big[\sup_{f\in\mathcal{H}_\lambda}|\mathcal{E}(f)-\mathcal{E}_{\mathbf{z}}(f)|\Big] \le 2\mathbb{E}_\rho\mathbb{E}_\varepsilon\Big[\sup_{f\in\mathcal{H}_\lambda}\frac{1}{m}\Big|\sum_{i=1}^m \varepsilon_i V(y_i,f(x_i))\Big|\Big] := 2R_m(\mathcal{V}_\lambda).$$

To handle $R_m(\mathcal{V}_\lambda)$, we apply Lemma 16. To this end, note that $\|f\|_\infty \le \kappa\sqrt{M/\lambda}$ for all $f\in\mathcal{H}_\lambda$, for fixed $\{y_i\}_{i=1}^m \in Y^m$. If we define functions

$$\phi_i(t) = \begin{cases} V(y_i,t)-V(y_i,0) & \text{when } |t|\le\kappa\sqrt{M/\lambda} \\ V(y_i,\kappa\sqrt{M/\lambda})-V(y_i,0) & \text{when } t\ge\kappa\sqrt{M/\lambda} \\ V(y_i,-\kappa\sqrt{M/\lambda})-V(y_i,0) & \text{when } t\le-\kappa\sqrt{M/\lambda}, \end{cases}$$

then $\phi_i(t) : \mathbb{R}\to\mathbb{R}$ has the Lipschitz constant $c_i = C_\lambda$ and $\phi_i(0) = 0$ for any i. Applying Lemma 16 to the space $\mathcal{H}_\lambda$, then $\mathbb{E}_\varepsilon\Big[\sup_{f\in\mathcal{H}_\lambda}\Big|\sum_{i=1}^m \varepsilon_i V(y_i,f(x_i))\Big|\Big]$ is bounded by

$$\begin{aligned} \mathbb{E}_\varepsilon\quad & \Big[\sup_{f\in\mathcal{H}_\lambda}\Big|\sum_{i=1}^m \varepsilon_i\phi_i(f(x_i))\Big|\Big] + \mathbb{E}_\varepsilon\Big[\sup_{f\in\mathcal{H}_\lambda}\Big|\sum_{i=1}^m \varepsilon_i V(y_i,0)\Big|\Big] \\ & \le 2\mathbb{E}_\varepsilon\Big[\sup_{f\in\mathcal{H}_\lambda}\Big|\sum_{i=1}^m C_\lambda\varepsilon_i f(x_i)\Big|\Big] + \Big[\mathbb{E}_\varepsilon\Big|\sum_{i=1}^m \varepsilon_i V(y_i,0)\Big|^2\Big]^{1/2} \\ & = 2C_\lambda\sqrt{\tfrac{M}{\lambda}}\,\mathbb{E}_\varepsilon\Big[\sup_{f\in\mathcal{H}}\Big|\sum_{i=1}^m \varepsilon_i f(x_i)\Big|\Big] + \Big(\mathbb{E}_\varepsilon\Big[\sum_{i,j=1}^m \varepsilon_i V(y_i,0)V(y_j,0)\varepsilon_j\Big]\Big)^{1/2}. \end{aligned}$$

It follows that

$$\begin{aligned} \mathbb{E}_\varepsilon\Big[\sup_{f\in\mathcal{H}_\lambda}\Big|\sum_{i=1}^m \varepsilon_i V(y_i,f(x_i))\Big|\Big] \quad & \le 2C_\lambda\sqrt{\tfrac{M}{\lambda}}\,\mathbb{E}_\varepsilon\Big[\sup_{f\in\mathcal{H}}\Big|\sum_{i=1}^m \varepsilon_i f(x_i)\Big|\Big] \\ & \quad + M\sqrt{m}. \end{aligned} \tag{25}$$

For the first term on the right hand side of the above inequality, we use the reproducing property (3) and obtain

$$\begin{aligned}\sup_{f\in\mathcal{H}}|\textstyle\sum_{i=1}^m \varepsilon_i f(x_i)| &= \sup_{\sigma\in\Sigma}\sup_{\substack{f\in\mathcal{H}_{K^\sigma}\\ \|f\|_{K^\sigma}\le 1}} |\langle \textstyle\sum_{i=1}^m \varepsilon_i K^\sigma_{x_i}, f\rangle_{K^\sigma}| = \sup_{\sigma\in\Sigma}\left\|\displaystyle\sum_{i=1}^m \varepsilon_i K^\sigma_{x_i}\right\|_{K^\sigma} \\ &= \left[\sup_{\sigma\in\Sigma}\textstyle\sum_{i,j=1}^m \varepsilon_i K^\sigma(x_i,x_j)\varepsilon_j\right]^{1/2}.\end{aligned} \tag{26}$$

But by the definition of the fundamental set $\mathcal{F}$, we know that

$$\begin{aligned}\sup_{\sigma\in\Sigma}\textstyle\sum_{i,j=1}^m \varepsilon_i K^\sigma(x_i,x_j)\varepsilon_j \le\ & m\sup_{\sigma\in\Sigma}\sup_{s\in X}|\displaystyle\sum_{i=1}^m \varepsilon_i K^\sigma(x_i,s)| \\ &= m\sup_{f\in\mathcal{F}}|\sum_{i=1}^m \varepsilon_i f(x_i)|.\end{aligned} \tag{27}$$

Combining the estimates (25), (26), and (27) implies that

$$\begin{aligned}R_m(\mathcal{V}_\lambda) &\le 2C_\lambda\sqrt{\tfrac{M}{\lambda}}\,\mathbb{E}_{\rho_X}\mathbb{E}_\varepsilon\Big[\tfrac{1}{m}\sup_{f\in\mathcal{F}}|\sum_{i=1}^m \varepsilon_i f(x_i)|\Big]^{1/2} + \frac{M}{\sqrt{m}} \\ &\le 2C_\lambda\sqrt{\tfrac{M}{\lambda}}\,\Big(R_m(\mathcal{F})\Big)^{1/2} + \tfrac{M}{\sqrt{m}}.\end{aligned}$$

This finishes Lemma 17. ■

The following error bound for the multi-kernel scheme (4) is a straightforward consequence of the sample error estimate and the error decomposition (13).

Theorem 18 *Let V be admissible with C_λ given by (24). Define $f_{\mathbf{z},\lambda}$ by (4). Then we have*

$$\mathbb{E}\Big[\mathcal{E}(f_{\mathbf{z},\lambda}) - \mathcal{E}(f_\rho^V)\Big] \le 4C_\lambda\sqrt{\frac{M}{\lambda}}\Big(R_m(\mathcal{F})\Big)^{1/2} + \frac{2M}{\sqrt{m}} + \widetilde{\mathcal{D}}(\lambda).$$

Proof By the error decomposition (13), a special choice $f_\lambda^V \in \mathcal{H}_{K^\sigma}$ with some $\sigma\in\Sigma$ defined by (12) gives us that

$$\mathcal{E}(f_{\mathbf{z},\lambda}) - \mathcal{E}(f_\rho^V) \le \Big\{\mathcal{E}(f_{\mathbf{z},\lambda}) - \mathcal{E}_{\mathbf{z}}(f_{\mathbf{z},\lambda}) + \mathcal{E}_{\mathbf{z}}(f_\lambda^V) - \mathcal{E}(f_\lambda^V)\Big\} + \widetilde{\mathcal{D}}(\lambda).$$

Together with the fact $\mathbb{E}(\mathcal{E}_{\mathbf{z}}(f_\lambda^V)) = \mathcal{E}(f_\lambda^V)$ and $f_{\mathbf{z},\lambda}\in\mathcal{H}_\lambda$ defined in (23), we know that

$$\mathbb{E}\Big[\mathcal{E}(f_{\mathbf{z},\lambda}) - \mathcal{E}_{\mathbf{z}}(f_{\mathbf{z},\lambda}) + \mathcal{E}_{\mathbf{z}}(f_\lambda^V) - \mathcal{E}(f_\lambda^V)\Big] \le \mathbb{E}\Big[\sup_{f\in\mathcal{H}_\lambda}|\mathcal{E}(f) - \mathcal{E}_{\mathbf{z}}(f)|\Big]$$

which in connection with Lemma 17 yields the desired estimate. ■

To estimate the Radmacher average $R_m(\mathcal{F})$ in Theorem 18, one can resort to the following bound using the empirical l^2 covering number, which is due to Dudley (1999); Van der Vaart and Weller (1996).

Lemma 19 *Let F be a class of uniformly bounded functions. Then there exists an absolute constant C such that, for any sample $\{x_i\}_{i=1}^m$, there holds*

$$\frac{1}{\sqrt{m}}\mathbb{E}_\varepsilon\Big[\sup_{f\in F}\big|\sum_{i=1}^m \varepsilon_i f(x_i)\big|\Big] \le C\int_0^{+\infty}\sqrt{\log\mathcal{N}_2(F,\mathbf{x},\varepsilon)}d\varepsilon.$$

Applying the estimate in the proof for Theorem 3, we can compute the Radmacher average $R_m(\mathcal{F})$ for Gaussian kernels with flexible variances, and hence yields the subsequent error bound.

Theorem 20 *Let X be a subset of $\mathbb{R}^n$ and V be admissible. Define $f_{\mathbf{z},\lambda}$ by (4) and the Gaussian kernels by (8). Then we have*

$$\mathbb{E}\Big[\mathcal{E}(f_{\mathbf{z},\lambda})-\mathcal{E}(f_\rho^V)\Big] \le C_\lambda\sqrt{\frac{C'M}{\lambda}}\Big(\frac{\log^2 m}{m}\Big)^{1/4}+\frac{2M}{\sqrt{m}}+\widetilde{\mathcal{D}}(\lambda), \tag{28}$$

where C' is a constant independent of m or λ.

Proof Recall the fundamental function set $\mathcal{F}$ defined by (7). Consider the larger set

$$\widetilde{\mathcal{F}} = \{K_x^\sigma = K^\sigma(x,\cdot) : \sigma\in\Sigma, x\in\mathbb{R}^n\}.$$

The estimate (22) tells us that for $0<\varepsilon<1$,

$$\begin{aligned}\log\mathcal{N}_2(\widetilde{\mathcal{F}},\mathbf{x},\varepsilon) &\le \log\mathcal{N}_\infty(\widetilde{\mathcal{F}},\mathbf{x},\varepsilon)\\ &\le n\log 2+\Big(n+\Big(\tfrac{16n}{\varepsilon}+2\Big)n\log\Big(\tfrac{2enm}{\varepsilon}\Big)\Big)\log\tfrac{4n^2m}{\varepsilon^2}.\end{aligned}$$

Since $K^\sigma\le 1$ for each $\sigma\in\Sigma$, we see that $\log\mathcal{N}_2(\widetilde{\mathcal{F}},\mathbf{x},\varepsilon)=0$ for any $\varepsilon\ge 1$. Applying Lemma 19, we have

$$R_m(\widetilde{\mathcal{F}})\le C''\frac{\log^2 m}{\sqrt{m}},$$

where C'' is an absolute constant independent of m. The trivial fact $R_m(\mathcal{F})\le R_m(\widetilde{\mathcal{F}})$ together with Theorem 18 gives us the desired result. ■

In order to get explicit error rates, we see from Theorems 18 and 20 that what left is to estimate the regularization error.

5.2 Regularization Error with Gaussians

In this subsection we exclusively focus on the multi-kernel regularization error (11) associated with the least square loss and Gaussian kernels. We show that how the Fourier analysis (Stein, 1970) can be applied to get the polynomial decay of the regularization error under Sobolev smoothness condition on the regression function.

Before we go to the main point, it is worth briefly mentioning why the multi-kernel regularization error can improve the error rates. To this end, note that, for the regularization error of a single Gaussian kernel, it was proved by Smale and Zhou (2003) that the polynomial decay $O(\lambda^s)$ for some $s>0$ is impossible under the Sobolev smoothness hypothesis on the regression function f_ρ. Actually, it only decays logarithmically $O((\log(1/\lambda))^{-s})$ for some $s>0$. Putting this logarithmical decay back into (28) and trading off λ and m, we notice that the error rate is unacceptably slow. Below we show that the multi-kernel scheme (4) associated with the least square loss and Gaussian kernels (9) with flexible isotropic variances can give regularization errors $\mathcal{D}(\lambda)$ of polynomial decays $O(\lambda^{\beta})$ for some $\beta > 0$, under the assumption of Sobolev smoothness on f_ρ. Combining this polynomial decay with (28) can give rise to tighter bounds compared to the single kernel case.

To estimate the multi-kernel regularization error (11), we introduce some basic facts about RKHS (Aronszajn, 1950). Let Ω be a domain (bounded or not) in $\mathbb{R}^n$. Let $\sigma > 0$. Consider the RKHS induced by the Gaussian kernel given in (9) as $K_\sigma(x,y) = \exp\{-\frac{|x-y|^2}{\sigma^2}\} : \Omega \times \Omega \to \mathbb{R}$. We denote it as $\mathcal{H}_{K_\sigma}(\Omega)$ with norm $\|\cdot\|_{\mathcal{H}_{K_\sigma}(\Omega)}$ for simplicity. Let $\widetilde{\Omega} \subset \Omega$. By restricting the kernel K_σ to $\tilde{\Omega} \times \tilde{\Omega}$, it also induces an RKHS in $\tilde{\Omega}$ denoted as $\mathcal{H}_{K_\sigma}(\tilde{\Omega})$. Then we know (Aronszajn, 1950) that

$$\mathcal{H}_{K_\sigma}(\tilde{\Omega}) = \Big\{ g = f|_{\tilde{\Omega}} : \ f \in \mathcal{H}_{K_\sigma}(\Omega) \Big\} \tag{29}$$

with norm

$$\|g\|_{\mathcal{H}_{K_\sigma}(\tilde{\Omega})} = \inf\{\|f\|_{\mathcal{H}_{K_\sigma}(\Omega)} : \ f|_{\tilde{\Omega}} = g\}. \tag{30}$$

Define the integral operator L_K associated with a Mercer kernel K and a Borel measure μ on Ω as

$$L_K f(x) := \int_\Omega K(x,t) f(t) d\mu(t), \quad x \in \Omega, \ f \in L^2_\mu(\Omega).$$

If Ω is a compact domain in $\mathbb{R}^n$, then L_K is a positive, self-adjoint, compact operator and its range lies in $C(\Omega)$. Take the square root $L_K^{1/2}$ of L_K, then

$$\|L_K^{1/2} f\|_K = \|f\|_{L^2_\mu(\Omega)} \qquad \forall f \in L^2_\mu(\Omega). \tag{31}$$

When $\Omega = \mathbb{R}^n$ and μ is the Lebesgue measure, we define for $\sigma > 0$

$$f^\sigma(x) = L_{K_\sigma} f(x) = \int_{\mathbb{R}^n} K_\sigma(x,y) f(y) dy, \quad x \in \mathbb{R}^n, \ f \in L^2(\mathbb{R}^n).$$

As in Steinwart and Scovel (2005), we shall use these functions as approximations of f_ρ to estimate the regularization error.

Lemma 21 *Let f^σ be defined for $f \in L^2(\mathbb{R}^n)$ as above. Then $f^\sigma \in \mathcal{H}_{K_\sigma}(\mathbb{R}^n)$ and*

$$\|f^\sigma\|_{\mathcal{H}_{K_\sigma}(\mathbb{R}^n)} \le (\sqrt{\pi}\sigma)^{n/2} \|f\|_{L^2(\mathbb{R}^n)}. \tag{32}$$

Proof We shall use notations and results on limits of reproducing kernels (see Theorem I in Section 9 of Aronszajn (1950)).

Denote E_j as the closed ball of $\mathbb{R}^n$ with radius j centered at zero. Then $\mathbb{R}^n = \bigcup_{j \in \mathbb{N}} E_j$. For $j \le j'$ and $f_{j'} \in \mathcal{H}_{K_\sigma}(E_{j'})$, the properties (29) and (30) of the restriction of RKHS tell us that

$$f_{j'}|_{E_j} \in \mathcal{H}_{K_\sigma}(E_j) \ \text{ and } \ \|f_{j'}|_{E_j}\|_{\mathcal{H}_{K_\sigma(E_j)}} \le \|f_{j'}\|_{\mathcal{H}_{K_\sigma}(E_{j'})}.$$

In order to show that $f^\sigma \in \mathcal{H}_{K_\sigma}(\mathbb{R}^n)$ and (32) holds, by Theorem I in Section 9 of Aronszajn (1950) it is sufficient to prove for $f_{\sigma,j} := f^\sigma|_{E_j}$ that

$$f_{\sigma,j} \in \mathcal{H}_{K_\sigma}(E_j) \quad \text{and} \quad \liminf_{j\to\infty} \|f_{\sigma,j}\|_{\mathcal{H}_{K_\sigma}(E_j)} \le (\sqrt{\pi}\sigma)^{n/2} \|f\|_{L^2(\mathbb{R}^n)}. \tag{33}$$

To this end, for $j \leq j'$, define

$$f_{\sigma,j,j'}(x) := \int_{E_{j'}} K_\sigma(x,y) f(y) dy \to f_{\sigma,j}(x) \quad \text{uniformly in} \quad C(E_j). \tag{34}$$

Then $f_{\sigma,j,j'} \in \mathcal{H}_{K_\sigma}(E_{j'})$ by (31) since $f \in L^2(E_{j'})$. Using (29) and (30) with $K_{\sigma,j'} := K_\sigma|_{E_{j'} \times E_{j'}}$, it yields

$$\|f_{\sigma,j,j'}|_{E_j}\|^2_{\mathcal{H}_{K_\sigma}(E_j)} \leq \|L_{K_{\sigma,j'}}(f|_{E_{j'}})\|^2_{\mathcal{H}_{K_\sigma}(E_{j'})}. \tag{35}$$

By (31), we have

$$\begin{aligned} \|L_{K_{\sigma,j'}}(f|_{E_{j'}})\|^2_{\mathcal{H}_{K_\sigma}(E_{j'})} &= \|L^{1/2}_{K_{\sigma,j'}}(f|_{E_{j'}})\|_{L^2(E_{j'})} = \langle L_{K_{\sigma,j'}}(f|_{E_{j'}}), f|_{E_{j'}} \rangle_{L^2(E_{j'})} \\ &\leq \|L_{K_{\sigma,j'}}(f|_{E_{j'}})\|_{L^2(E_{j'})} \|f|_{E_{j'}}\|_{L^2(E_{j'})}. \end{aligned} \tag{36}$$

Note that for each $x \in E_{j'}$, there holds $\int_{E_{j'}} K_\sigma(x,t)dt \leq \int_{\mathbb{R}^n} K_\sigma(x,t)dt = (\sqrt{\pi}\sigma)^n$. We get by the Schwarz inequality

$$\begin{aligned} \|L_{K_{\sigma,j'}} f\|^2_{L^2(E_{j'})} &= \int_{E_{j'}} \left| \int_{E_{j'}} K_\sigma(x,t) f(t) dt \right|^2 dx \\ &\leq \int_{E_{j'}} \left\{ \int_{E_{j'}} K_\sigma(x,t) dt \right\} \left\{ \int_{E_{j'}} K_\sigma(x,t) |f(t)|^2 dt \right\} dx \\ &\leq (\sqrt{\pi}\sigma)^n \int_{E_{j'}} |f(t)|^2 \left\{ \int_{E_{j'}} K_\sigma(x,t) dx \right\} dt \\ &\leq (\sqrt{\pi}\sigma)^{2n} \|f\|^2_{L^2(\mathbb{R}^n)}. \end{aligned}$$

Putting this estimate into (36), it follows from (35) that

$$\|f_{\sigma,j,j'}|_{E_j}\|_{\mathcal{H}_{K_\sigma}(E_j)} \leq \|L_{K_{\sigma,j'}}(f|_{E_{j'}})\|_{\mathcal{H}_{K_\sigma}(E_{j'})} \leq (\sqrt{\pi}\sigma)^{n/2} \|f\|_{L^2(\mathbb{R}^n)}.$$

Since the fixed ball of $\mathcal{H}_{K_\sigma}(E_j)$ with radius $(\sqrt{\pi}\sigma)^{n/2}\|f\|_{L^2(\mathbb{R}^n)}$ centered at zero is weakly compact, there exists a subsequence $\{j'_\ell\}_{\ell \in \mathbb{N}}$ of $\{j'\}$ such that

$$f_{\sigma,j,j'_\ell}|_{E_j} \rightharpoonup f^* \qquad \text{in } \mathcal{H}_{K_\sigma}(E_j) \quad \text{as} \quad \ell \to \infty.$$

Therefore

$$\begin{aligned} \|f^*\|^2_{\mathcal{H}_{K_\sigma}(E_j)} &= \lim_{\ell \to \infty} \langle f^*, f_{\sigma,j,j'_\ell}|_{E_j} \rangle_{\mathcal{H}_{K_\sigma}(E_j)} \\ &\leq \|f^*\|_{\mathcal{H}_{K_\sigma}(E_j)} \liminf_{\ell \to \infty} \|f_{\sigma,j,j'_\ell}|_{E_j}\|_{\mathcal{H}_{K_\sigma}(E_j)} \end{aligned}$$

which tells us that

$$\|f^*\|_{\mathcal{H}_{K_\sigma}(E_j)} \leq \liminf_{\ell \to \infty} \|f_{\sigma,j,j'_\ell}|_{E_j}\|_{\mathcal{H}_{K_\sigma}(E_j)} \leq (\sqrt{\pi}\sigma)^{n/2} \|f\|_{L^2(\mathbb{R}^n)}. \tag{37}$$

By the reproducing property (3), we also have, for each $x \in E_j$

$$f_{\sigma,j,j'_\ell}|_{E_j}(x) = \langle f_{\sigma,j,j'_\ell}|_{E_j}, K_{\sigma,j}(x,\cdot) \rangle_{\mathcal{H}_{K_\sigma}(E_j)} \to \langle f^*, K_{\sigma,j}(x,\cdot) \rangle_{\mathcal{H}_{K_\sigma}(E_j)} = f^*(x)$$

which in connection with (34) gives us that

$$f_{\sigma,j} = f^* \in \mathcal{H}_{K_\sigma}(E_j).$$

Together with (37) and (33), we know that Lemma 21 holds true. ■

Proposition 22 *Let X be a domain in $\mathbb{R}^n$ with Lipschitz boundary. Suppose $f_\rho \in H^s(X)$ for some $s > 0$. Consider the multi-kernel scheme (4) with the least square loss $V(y,t) = (y-t)^2$ and the Gaussian kernels (9) with $\Sigma = (0,\infty)$. Then, the following statements hold true.*

(1) If $n/2 < s \le n/2+2$, then there holds

$$\widetilde{\mathcal{D}}(\lambda) \le \inf_{\sigma\in(0,\infty)} \inf_{f\in\mathcal{H}_{K_\sigma}} \Big\{ \|f-f_\rho\|^2_{C(X)} + \lambda \|f\|^2_{K_\sigma} \Big\}.$$

For any $0<\varepsilon<2s-n$, there exists a constant $C_{\varepsilon,s,X}$ such that

$$\widetilde{\mathcal{D}}(\lambda) \le C_{\varepsilon,s,X} \|f_\rho\|^2_{H^s(X)} \, \lambda^{\frac{2s-\varepsilon-n}{2s-\varepsilon}}.$$

(2) If X is bounded, ρ_X is the Lebesgue measure on X and $s\le 2$, then there exists a constant $C_{s,n,X}$ independent of λ such that

$$\widetilde{\mathcal{D}}(\lambda) \le C_{s,n,X} \|f_\rho\|^2_{H^s(X)} \, \lambda^{\frac{2s}{2s+n}}. \tag{38}$$

Proof For the least square loss, we have

$$\mathcal{E}(f) - \mathcal{E}(f_\rho) = \|f-f_\rho\|^2_{L^2_{\rho_X}(X)}. \tag{39}$$

Since X has a Lipschitz boundary, we know (Stein, 1970) that there exists an extension function $\widetilde{f_\rho} \in H^s(\mathbb{R}^n)$ and an absolute constant $C_{s,X}$ such that

$$\widetilde{f_\rho}|_X = f_\rho \quad \text{and} \quad \|\widetilde{f_\rho}\|_{H^s(\mathbb{R}^n)} \le C_{s,X}\|f_\rho\|_{H^s(X)}. \tag{40}$$

Define the normalized kernel $\widetilde{K}_\sigma = (\sqrt{\pi}\sigma)^{-n} K_\sigma$. Let

$$f^\sigma_\rho(x) = (\sqrt{\pi}\sigma)^{-n} L_{K_\sigma}(\widetilde{f_\rho})(x) = \int_{\mathbb{R}^n} \widetilde{K}_\sigma(x,y)\widetilde{f_\rho}(y)dy, \qquad x\in\mathbb{R}^n.$$

Then we know that f^σ_ρ belongs to $\mathcal{H}_{K_\sigma}(\mathbb{R}^n)$ by Lemma 21. Combined with the fact (29), it follows that $g^\sigma_\rho := f^\sigma_\rho|_X \in \mathcal{H}_{K_\sigma}(X)$. Take $f = g^\sigma_\rho$ in the definition of the regularization error $\widetilde{\mathcal{D}}(\lambda)$. We see by (39) that

$$\widetilde{\mathcal{D}}(\lambda) \le \inf_{\sigma\in(0,\infty)} \{ \|g^\sigma_\rho - f_\rho\|^2_{L^2_{\rho_X}(X)} + \lambda \|g^\sigma_\rho\|^2_{K_\sigma} \}. \tag{41}$$

By the relations (29) and (30) on the restriction of RKHS and Lemma 21, we see that

$$\|g^\sigma_\rho\|_{K_\sigma} \le \|f^\sigma_\rho\|_{\mathcal{H}_{K_\sigma}(\mathbb{R}^n)} = (\sqrt{\pi}\sigma)^{-n} \|L_{K_\sigma}\widetilde{f_\rho}\|_{\mathcal{H}_{K_\sigma}(\mathbb{R}^n)} \le (\sqrt{\pi}\sigma)^{-n/2} \|\widetilde{f_\rho}\|_{L^2(\mathbb{R}^n)}.$$

Together with (40), it yields

$$\|g^\sigma_\rho\|_{K_\sigma} \le C_{s,X} (\sqrt{\pi}\sigma)^{-n/2} \|f_\rho\|_{H^s(X)}. \tag{42}$$

To bound the right hand side of (41), we need the Fourier transform defined for $f\in L^1(\mathbb{R}^n)$ as

$$\widehat{f}(\xi) = \int_{\mathbb{R}^n} f(x) e^{-ix\cdot\xi} dx, \qquad \xi\in\mathbb{R}^n.$$

It has a natural extension to $L^2(\mathbb{R}^n)$ satisfying $\|\widehat{f}\|_{L^2(\mathbb{R}^n)} = (2\pi)^{n/2}\|f\|_{L^2(\mathbb{R}^n)}$ (the Plancherel formula). The norm for functions in the Sobolev space $H^s(\mathbb{R}^n)$ can be expressed as $\|f\|_{H^s(\mathbb{R}^n)} = (2\pi)^{-n/2}\left(\int_{\mathbb{R}^n}(1+|\xi|^2)^s|\widehat{f}(\xi)|^2 d\xi\right)^{1/2}$. One nice property of the Fourier transform says that the Fourier transform of the convolution $f*g(x) = \int_{\mathbb{R}^n} f(x-y)g(y)dy$ equals $\widehat{f}(\xi)\widehat{g}(\xi)$. It implies that $\widehat{f_\rho^\sigma}(\xi) = e^{-\sigma^2|\xi|^2/4}\widehat{\widetilde{f}}_\rho(\xi)$ since the Fourier transform of the function $(\sqrt{\pi}\sigma)^{-n}e^{-|x|^2/\sigma^2}$ is $e^{-\sigma^2|\xi|^2/4}$.

(1) For any marginal distribution ρ_X, there holds $\|f\|_{L^2_{\rho_X}} \leq \|f\|_{C(X)}$. Then the first inequality follows from (39).

Since $X \subseteq \mathbb{R}^n$ and $(n+\varepsilon)/2 > n/2$, we know that the Sobolev space $H^{(n+\varepsilon)/2}(\mathbb{R}^n)$ can be embedded into $C(\mathbb{R}^n)\cap L^\infty(\mathbb{R}^n)$, there exists a constant $C'_{\varepsilon,X}$ such that $\|f\|_{L^\infty(\mathbb{R}^n)} \leq C'_{\varepsilon,X}\|f\|_{H^{(n+\varepsilon)/2}(\mathbb{R}^n)}$. It follows that

$$\|g_\rho^\sigma - f_\rho\|_{C(X)} \leq \|f_\rho^\sigma - \widetilde{f}_\rho\|_{L^\infty(\mathbb{R}^n)} \leq C'_{\varepsilon,X}\|f_\rho^\sigma - \widetilde{f}_\rho\|_{H^{(n+\varepsilon)/2}(\mathbb{R}^n)}.$$

Write

$$\|f_\rho^\sigma - \widetilde{f}_\rho\|^2_{H^{(n+\varepsilon)/2}(\mathbb{R}^n)} = (2\pi)^{-n}\int_{\mathbb{R}^n}(1+|\xi|^2)^{(n+\varepsilon)/2}\left|(e^{-\sigma^2|\xi|^2/4}-1)\widehat{\widetilde{f}}_\rho(\xi)\right|^2 d\xi.$$

Since $\frac{s}{2} - \frac{n+\varepsilon}{4} < 1$, we have $|e^{-\sigma^2|\xi|^2/4} - 1| \leq (\sigma^2|\xi|^2/4)^{\frac{s}{2}-\frac{n+\varepsilon}{4}}$. Hence

$$\begin{aligned}\|f_\rho^\sigma - \widetilde{f}_\rho\|^2_{H^{(n+\varepsilon)/2}(\mathbb{R}^n)} &\leq \frac{\sigma^{2s-(n+\varepsilon)}}{(2\pi)^n}\int_{\mathbb{R}^n}(1+|\xi|^2)^{(n+\varepsilon)/2}(|\xi|^2)^{s-(n+\varepsilon)/2}|\widehat{\widetilde{f}}_\rho(\xi)|^2 d\xi \\ &\leq \frac{\sigma^{2s-(n+\varepsilon)}}{(2\pi)^n}\int_{\mathbb{R}^n}(1+|\xi|^2)^s|\widehat{\widetilde{f}}_\rho(\xi)|^2 d\xi = \sigma^{2s-(n+\varepsilon)}\|\widetilde{f}_\rho\|^2_{H^s(\mathbb{R}^n)}.\end{aligned}$$

In connection with (40), this implies that

$$\|g_\rho^\sigma - f_\rho\|_{C(X)} \leq C'_{\varepsilon,X}\sigma^{s-(n+\varepsilon)/2}C_{s,X}\|f_\rho\|_{H^s(X)}.$$

Combining with (42) and choosing $\sigma = \lambda^{1/(2s-\varepsilon)}$ this proves the first statement of the proposition.

(2) If X is bounded and ρ_X is the Lebesgue measure on X, then $\|g_\rho^\sigma - f_\rho\|^2_{L^2_{\rho_X}} = \|g_\rho^\sigma - f_\rho\|^2_{L^2(X)}$ and by the Plancherel formula,

$$\|g_\rho^\sigma - f_\rho\|^2_{L^2(X)} \leq \|f_\rho^\sigma - \widetilde{f}_\rho\|^2_{L^2(\mathbb{R}^n)} = (2\pi)^{-n}\int_{\mathbb{R}^n}\left|(e^{-\sigma^2|\xi|^2/4}-1)\widehat{\widetilde{f}}_\rho(\xi)\right|^2 d\xi.$$

Observe from the restriction $s \leq 2$ that $1-e^{-t} \leq (1-e^{-t})^{s/2} \leq t^{s/2}$ for $t > 0$. Applying this to $t = \sigma^2|\xi|^2/4$ we obtain

$$\begin{aligned}\|g_\rho^\sigma - f_\rho\|^2_{L^2(X)} &\leq \frac{\sigma^{2s}}{(2\pi)^n 4^s}\int_{\mathbb{R}^n}|\xi|^{2s}|\widehat{\widetilde{f}}_\rho(\xi)|^2 d\xi \\ &\leq \sigma^{2s}\|\widetilde{f}_\rho\|^2_{H^s(\mathbb{R}^n)} \leq C^2_{s,X}\sigma^{2s}\|f_\rho\|^2_{H^s(X)},\end{aligned}$$

where we have used (40) in the last inequality. Putting this estimate and (42) into (41) gives us the second statement of the proposition corresponding to the choice $\sigma = \lambda^{1/(2s+n)}$. ■

Finally, we are able to derive error rates for the multi-kernel regularization scheme (4) associated with Gaussian kernels (8) with flexible variances. This is done by putting the improved regularization error bound in Proposition 22 into the total error bound in Theorem 20. We demonstrate the approach for regression with least square loss and for classification with hinge loss.

6. Error Rates with Gaussians

In this section we prove Examples 1 and 2. First, let us consider Example 1, that is, the case of the least square regularized regression: $Y = \mathbb{R}$. If $|y| \leq M_0$ almost surely, the least square loss $V(y,s) = (y-s)^2$ is admissible with respect to ρ and $M = \|V(y,0)\|_{L^\infty_\rho(Z)} \leq M_0^2$, $C_\lambda \leq 2M_0(1+\kappa/\sqrt{\lambda})$. By Theorem 20 and the special property (39) of the loss function, we immediately get the following result.

Proposition 23 *Let $V(y,t) = (y-t)^2$, $X \subseteq \mathbb{R}^n$ and $\{K^\sigma\}$ be given by (8). Define $f_{\mathbf{z},\lambda}$ by (14). If $0 < \lambda \leq 1$, then there exists a constant $\widetilde{C}$ independent of m, λ such that*

$$\mathbb{E}\Big[\|f_{\mathbf{z},\lambda} - f_\rho\|_\rho^2\Big] \leq \widetilde{C}\Big(\frac{\log^2 m}{m\lambda^4}\Big)^{1/4} + \widetilde{\mathcal{D}}(\lambda). \tag{43}$$

Using this proposition, we can provide the proof of Example 1.
Proof of Example 1. Since the set (8) of Gaussian kernels with general variances $\sigma = (\sigma_1, \ldots, \sigma_n) \in (0,\infty)^n$ contains the set (9) of Gaussian kernels with isotropic variances, we see that

$$\widetilde{\mathcal{D}}(\lambda) \leq \inf_{\sigma\in(0,\infty)} \inf_{f\in\mathcal{H}_{K_\sigma}} \Big\{\mathcal{E}(f) - \mathcal{E}(f_\rho) + \lambda\|f\|_{K_\sigma}^2\Big\}.$$

(1) When $n/2 < s \leq n/2+2$ and $0 < \varepsilon < 2s-n$, we know from Proposition 22 that

$$\widetilde{\mathcal{D}}(\lambda) \leq C_{\varepsilon,s,X}\|f_\rho\|^2_{L^2_{\rho_X}(X)}\lambda^{\frac{2s-\varepsilon-n}{2s-\varepsilon}}.$$

Putting this into (43) and choosing $\lambda = m^{-\frac{2s-\varepsilon}{4(4s-n-2\varepsilon)}}$ verifies the first error estimate in Example 1.

(2) If X is bounded, ρ_X is the Lebesgue measure on X, and $s \leq 2$, we apply the bound (38) in Proposition 22. Together with the above inequality, we know that

$$\widetilde{\mathcal{D}}(\lambda) \leq C_{s,n,X}\|f_\rho\|^2_{L^2_{\rho_X}(X)}\lambda^{\frac{2s}{2s+n}}.$$

In connection with the error bound (43) we see that when $\lambda = m^{-\frac{2s+n}{4(4s+n)}}$, the error estimate in Part (2) holds true. ■

Now we move on to establish Example 2 for the regularized classification with the hinge loss $V(y,s) = (1-ys)_+$. In this case we take $Y = \{1,-1\}$. It is easy to see that V is admissible with $M = 1$ and $C_\lambda = 1$. An important relation between the excess misclassification error and the excess error was given by Zhang (2004) as

$$\mathcal{R}(\mathrm{sgn}(f)) - \mathcal{R}(f_c) \leq \mathcal{E}(f) - \mathcal{E}(f_c), \qquad \forall f: X \to \mathbb{R}.$$

Then the following result is an easy consequence of Theorem 20.

Proposition 24 *Let $Y = \{1,-1\}$, $V(y,t) = (1-yt)_+$, $X \subseteq \mathbb{R}^n$ and $\{K^\sigma\}$ be given by (8). Define $f_{\mathbf{z},\lambda}$ by (15). If $0 < \lambda \leq 1$, then there exists a constant C' independent of m, λ such that*

$$\mathbb{E}\Big[\mathcal{R}(sgn(f_{\mathbf{z},\lambda})) - \mathcal{R}(f_c)\Big] \leq \sqrt{\frac{C'}{\lambda}}\Big(\frac{\log^2 m}{m}\Big)^{1/4} + \widetilde{\mathcal{D}}(\lambda).$$

We are in a position to prove Example 2 by Proposition 24.

Proof of Example 2. It was shown by Chen et al. (2004) that if ρ is separable with some exponent $\theta > 0$ then there exists some constant c' such that $\widetilde{\mathcal{D}}(\lambda) \leq c'\lambda^{\frac{\theta}{2+\theta}}$. Choosing $\lambda = m^{-\frac{2+\theta}{2(2+3\theta)}}$ gives us the desired result. ■

Acknowledgments

We would like to thank the referees for their constructive suggestions and comments. This work was supported by the Research Grants Council of the Hong Kong [Project No. CityU 103704], City University of Hong Kong [Project No. 7001983], and the National Science Fund for Distinguished Young Scholars of China [Project No. 10529101]. The corresponding author is Ding-Xuan Zhou. Yiming Ying is on leave from Institute of Mathematics, Chinese Academy of Science, Beijing 100080, P. R. China.

Appendix A.

This appendix includes complete proofs of two lemmas which are essentially proved in Alon et al. (1997); Ledoux and Talagrand (1991) with slightly different forms.

Proof of Lemma 12. Part (a) is an easy consequence of Lemma 2.4 in Alon et al. (1997).

Set $d_0 = \min\{m, d\}$. As in the proof of Lemma 3.5 of Alon et al. (1997), one can bound the empirical covering number by packing numbers which can then be estimated by Lemma 3.3 in Alon et al. (1997) as

$$\sup_{\mathbf{x}\in X^m} \mathcal{N}_\infty(\mathcal{G}, \mathbf{x}, \varepsilon) \leq 2\left(m\left(\frac{2}{\varepsilon}\right)^2\right)^{\log y + 1},$$

where

$$y = \sum_{i=1}^{d_0} \binom{m}{i} \left(\frac{2}{\varepsilon}\right)^i \leq \left(\frac{2}{\varepsilon}\right)^{d_0} \left(\frac{em}{d_0}\right)^{d_0} \leq \left(\frac{2em}{\varepsilon}\right)^{d_0} \leq \left(\frac{2em}{\varepsilon}\right)^d.$$

This verifies Part (b). ■

Proof of Lemma 16. The first statement is immediate from the definition of Rademacher averages.

For the second statement, we use Theorem 4.12 in Ledoux and Talagrand (1991). It tells us the following result: If T is a bounded subset of $\mathbb{R}^m$, each function ϕ_i with the Lipschitz constant not more than 1 satisfies $\phi_i(0) = 0$, and a function $G : \mathbb{R}_+ \to \mathbb{R}_+$ is convex and nondecreasing, then there holds

$$\mathbb{E}_\varepsilon G\Big(\frac{1}{2} \sup_{t\in T} \Big|\sum_{i=1}^m \varepsilon_i \phi_i(t_i)\Big|\Big) \leq \mathbb{E}_\varepsilon G\Big(\sup_{t\in T} \Big|\sum_{i=1}^m \varepsilon_i t_i\Big|\Big). \tag{44}$$

Fixed $\{x_i\}_{i=1}^m$. Then $T := \{(c_1 f(x_1), \cdots, c_m f(x_m)) : f \in F\}$ is a bounded subset of $\mathbb{R}^m$ since F is uniformly bounded. Applying (44) with $G(u) = u$ and $\widetilde{\phi}_i(x) = \phi_i(x/c_i)$, we have

$$\begin{aligned} \mathbb{E}_\varepsilon\Big[\sup_{f\in F} \Big|\sum_{i=1}^m \varepsilon_i \phi_i(f(x_i))\Big|\Big] &= \mathbb{E}_\varepsilon\Big[\sup_{t\in T} \Big|\sum_{i=1}^m \varepsilon_i \widetilde{\phi}_i(t_i)\Big|\Big] \leq 2\mathbb{E}_\varepsilon\Big[\sup_{t\in T} \Big|\sum_{i=1}^m \varepsilon_i t_i\Big|\Big] \\ &= 2\mathbb{E}_\varepsilon\Big[\sup_{f\in F} \Big|\sum_{i=1}^m c_i \varepsilon_i f(x_i)\Big|\Big] \end{aligned}$$

which proves our second statement. ■

Appendix B.

The arguments in this paper do not change much if we replace a minimizer of the optimization problem (4) by an ε-*minimizer* $f^{\varepsilon}_{\mathbf{z},\lambda} \in \mathcal{H}_{K^\sigma}$ (with some $\sigma \in \Sigma$) satisfying

$$\frac{1}{m}\sum_{i=1}^{m} V(y_i, f^{\varepsilon}_{\mathbf{z},\lambda}(x_i)) + \lambda \|f^{\varepsilon}_{\mathbf{z},\lambda}\|^2_{K^\sigma} \le \min_{\sigma\in\Sigma}\min_{f\in\mathcal{H}_{K^\sigma}} \left\{\frac{1}{m}\sum_{i=1}^{m} V(y_i, f(x_i)) + \lambda\|f\|^2_{K^\sigma}\right\} + \varepsilon.$$

The existence of an ε-minimizer for any given $0 < \varepsilon \le 1$ can be seen in Wu et al. (2007) where it was shown that $\min_{f\in\mathcal{H}_{K^\sigma}}\{\frac{1}{m}\sum_{i=1}^{m} V(y_i, f(x_i)) + \lambda\|f\|^2_{K^\sigma}\}$ is continuous as a function of $\sigma \in \Sigma$ if $K^\sigma(x,x')$ is continuous with respect to $\sigma\in\Sigma$ for each fixed pair $(x,x') \in X\times X$.

In this appendix, we verify the existence for the scheme (4) involving the least-square loss and Gaussians (9) with flexible variances under some mild conditions on the sample $\mathbf{z}$. Note that the shortest distance between pairs of distinct points from $\{x_1,\ldots,x_m\}$ is achieved by some pair (x_{i_1}, x_{i_2}) with $i_1 \ne i_2$. The condition for our existence result is that such a minimizing pair is unique and the sample values y_{i_1}, y_{i_2} have the same sign. Since x_{i_1} and x_{i_2} are close, this assumption of having the same sign for the sample values is reasonable.

Proposition 25 *Let $\lambda > 0$, X be a compact subset of $\mathbb{R}^n$ and $K_\sigma(x,y) = \exp\{-\frac{|x-y|^2}{\sigma^2}\}$ for $x,y\in X$. Consider the scheme with $0 < b < \infty$*

$$f_{\mathbf{z},\lambda} := \arg\min_{\sigma\in(0,b]}\min_{f\in\mathcal{H}_{K^\sigma}}\left\{\frac{1}{m}\sum_{i=1}^{m}(f(x_i)-y_i)^2 + \lambda\|f\|^2_{K^\sigma}\right\}. \tag{45}$$

If we can find $\{i_1 \ne i_2\} \subset \{1,\ldots,m\}$ such that

$$y_{i_1}y_{i_2} > 0 \quad \text{and} \quad |x_{i_1} - x_{i_2}| \le |x_i - x_j| \text{ for any } \{i\ne j\}\subset\{1,\ldots,m\}$$

with equality valid only for $(i,j) = (i_1,i_2)$ or (i_2,i_1), then the existence of a solution to (45) holds true.

Proof For $\sigma > 0$ we denote

$$e_{\mathbf{z},\lambda}(\sigma) = \min_{f\in\mathcal{H}_{K^\sigma}}\left\{\frac{1}{m}\sum_{i=1}^{m}(f(x_i)-y_i)^2 + \lambda\|f\|^2_{K^\sigma}\right\}.$$

The minimizer of this one-layer optimization problem exists, is unique, and can be expressed as $f^\sigma_{\mathbf{z},\lambda} = \sum_{i=1}^m c^\sigma_i K^\sigma_{x_i}$. The coefficient vector $c^\sigma := (c^\sigma_i)_{i=1}^m$ is the solution of the linear system

$$([K^\sigma]_{\mathbf{x}} + m\lambda I_m)\,c = \mathbf{y}$$

where $\mathbf{y}$ is the vector $(y_i)_{i=1}^m$ and $[K^\sigma]_{\mathbf{x}}$ is the $m\times m$ matrix $(K^\sigma(x_i,x_j))_{i,j=1}^m$. Each main diagonal entry of $[K^\sigma]_{\mathbf{x}}$ is 1. A simple computation yields $\frac{1}{m}\sum_{i=1}^m(f^\sigma_{\mathbf{z},\lambda}(x_i) - y_i)^2 = m\lambda^2\|c^\sigma\|^2, \|f^\sigma_{\mathbf{z},\lambda}\|^2_{K^\sigma} = (c^\sigma)^T[K^\sigma]_{\mathbf{x}}c^\sigma$ and

$$e_{\mathbf{z},\lambda}(\sigma) = \lambda\mathbf{y}^T\left([K^\sigma]_{\mathbf{x}} + m\lambda I_m\right)^{-1}\mathbf{y}.$$

In the following, we will show that $\lim_{\sigma\to 0} e_{\mathbf{z},\lambda}(\sigma)$ is strictly larger than $e_{\mathbf{z},\lambda}(\sigma)$ for any $\sigma\in(0,\sigma_2]$ with some $\sigma_2 > 0$. Thereby, the minimizer of $e_{\mathbf{z},\lambda}$ should be achieved at a positive number in $(0,b]$ since $e_{\mathbf{z},\lambda}(\cdot)$ is a continuous function.

Without loss of generality, we assume that $i_1 = 1, i_2 = 2$. Denote $a_\sigma = \exp\{-\frac{|x_1-x_2|^2}{\sigma^2}\}$. We see that $a_\sigma > 0$ and $\lim_{\sigma\to 0} a_\sigma = 0$. Our assumption on the distances $|x_i - x_j| \geq |x_1 - x_2|$ $(i \neq j)$ tells us that the off-diagonal entries of $[K^\sigma]_{\mathbf{x}}$ decays to 0 faster than a_σ except for the two entries at $(1,2)$ and $(2,1)$. That is,

$$[K^\sigma]_{\mathbf{x}} + m\lambda I_m = A_\sigma + a_\sigma B_\sigma, \quad A_\sigma := \begin{bmatrix} m\lambda+1 & a_\sigma & 0 \\ a_\sigma & m\lambda+1 & 0 \\ 0 & 0 & (m\lambda+1)I_{m-2} \end{bmatrix}$$

and $\lim_{\sigma\to 0} B_\sigma = 0$. The inverse of A_σ has a nice form

$$A_\sigma^{-1} = \begin{bmatrix} \frac{m\lambda+1}{(m\lambda+1)^2-a_\sigma^2} & -\frac{a_\sigma}{(m\lambda+1)^2-a_\sigma^2} & 0 \\ -\frac{a_\sigma}{(m\lambda+1)^2-a_\sigma^2} & \frac{m\lambda+1}{(m\lambda+1)^2-a_\sigma^2} & 0 \\ 0 & 0 & \frac{1}{m\lambda+1}I_{m-2} \end{bmatrix}.$$

Recall that the norm of an $m \times m$ matrix A is defined by $\|A\| = \sup\{\|Ax\| : \|x\| = 1\}$. If in addition, A is symmetric, then $\|A\| = \max\{|\lambda_i| : i = 1,2,\ldots,m\}$ where $\{\lambda_i : i = 1,2,\cdots,m\}$ are the eigenvalues of A. Hence,

$$\|A_\sigma^{-1}\| \leq \frac{1}{m\lambda + 1 - a_\sigma} \leq \frac{1}{m\lambda}.$$

Moreover, since $\lim_{\sigma\to 0} B_\sigma = 0$, there exists $0 < \sigma_1 \leq b$ such that

$$\|A_\sigma^{-1/2} B_\sigma A_\sigma^{-1/2}\| \leq 1 \quad \forall 0 < \sigma \leq \sigma_1.$$

Therefore, for any $0 < \sigma \leq \sigma_1$ there holds

$$\|(I + a_\sigma A_\sigma^{-1/2} B_\sigma A_\sigma^{-1/2})^{-1}\| \leq \frac{1}{1-a_\sigma}.$$

If we write $([K^\sigma]_{\mathbf{x}} + m\lambda I_m)^{-1}$ as $A_\sigma^{-1/2}(I + a_\sigma A_\sigma^{-1/2} B_\sigma A_\sigma^{-1/2})^{-1} A_\sigma^{-1/2}$, then

$$([K^\sigma]_{\mathbf{x}} + m\lambda I_m)^{-1} - A_\sigma^{-1} = -a_\sigma A_\sigma^{-1/2}(I + a_\sigma A_\sigma^{-1/2} B_\sigma A_\sigma^{-1/2})^{-1} A_\sigma^{-1/2} B_\sigma A_\sigma^{-1}.$$

Consequently,

$$\|([K^\sigma]_{\mathbf{x}} + m\lambda I_m)^{-1} - A_\sigma^{-1}\| \leq \frac{a_\sigma}{1-a_\sigma}\Big(\frac{1}{m\lambda}\Big)^2 \|B_\sigma\|.$$

This implies that

$$e_{\mathbf{z},\lambda}(\sigma) \leq \lambda \mathbf{y}^T A_\sigma^{-1}\mathbf{y} + \lambda\frac{a_\sigma\|B_\sigma\|}{1-a_\sigma}\Big(\frac{1}{m\lambda}\Big)^2.$$

But

$$\lambda \mathbf{y}^T A_\sigma^{-1}\mathbf{y} = \frac{\lambda}{m\lambda+1}\|\mathbf{y}\|^2 + \frac{\lambda a_\sigma^2(y_1^2+y_2^2)}{(m\lambda+1)((m\lambda+1)^2-a_\sigma^2)} - \frac{2\lambda a_\sigma y_1 y_2}{(m\lambda+1)^2-a_\sigma^2},$$

which means that $e_{\mathbf{z},\lambda}(\sigma) - \frac{\lambda}{m\lambda+1}\|\mathbf{y}\|^2$ is bounded by

$$\lambda a_\sigma\Big[\frac{\|B_\sigma\|}{1-a_\sigma}\Big(\frac{1}{m\lambda}\Big)^2 + \frac{a_\sigma(y_1^2+y_2^2)}{(m\lambda+1)((m\lambda+1)^2-a_\sigma^2)} - \frac{2y_1y_2}{(m\lambda+1)^2-a_\sigma^2}\Big].$$

Using the properties that $y_1 y_2 > 0$, $\lim_{\sigma\to 0} a_\sigma = 0$ and $\lim_{\sigma\to 0} \|B_\sigma\| = 0$ again, we know there exists $0 < \sigma_2 \le \sigma_1$ such that

$$e_{\mathbf{z},\lambda}(\sigma) < \frac{\lambda}{m\lambda + 1}\|\mathbf{y}\|^2, \qquad \forall 0 < \sigma \le \sigma_2.$$

We also observe that $\lim_{\sigma\to 0}[K^\sigma]_{\mathbf{x}} + m\lambda I_m = (m\lambda + 1)I_m$. Hence

$$\lim_{\sigma\to 0} e_{\mathbf{z},\lambda}(\sigma) = \frac{\lambda}{m\lambda + 1}\|\mathbf{y}\|^2 > e_{\mathbf{z},\lambda}(\sigma) \qquad \forall 0 < \sigma \le \sigma_2.$$

It means that the infimum in (45) cannot be achieved as $\sigma \to 0$. By the continuity of $e_{\mathbf{z},\lambda}(\cdot)$, the existence of a solution to (45) follows from that of the optimization problem for σ lying in a compact subset of $(0, b]$ proved in Wu et al. (2007). ■

The above existence result largely depends on the least square loss and the assumption on the data. It remains an open problem on how to prove the existence of the minimizer of the multi-kernel scheme (4) associated with general loss functions and data.

References

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

N. Alon, S. Ben-David, S. N. Cesa-Bianchi, and D. Haussler. Scale-sensitive dimensions, uniform convergence and learnability. *Journal of the ACM*, 44:615–631, 1997.

P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101:138–156, 2006.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

N. Cristianini, J. Shawe-Taylor, and C. Campbell. Dynamically adapting kernels in support vector machines. In *Advances in Neural Information Processing Systems* 11 (M. S. Kearns, S. A. Solla, and D. A. Cohn, eds), MIT Press, 1999.

D. R. Chen, Q. Wu, Y. Ying, and D. X. Zhou. Support vector machine soft margin classifiers: Error analysis. *Journal of Machine Learning Research*, 5:1143–1175, 2004.

O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46:131–159, 2002.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

F. Cucker and S. Smale. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39:1–49, 2001.

F. Cucker and D. X. Zhou. *Learning Theory: An Approximation Theory Viewpoint*. Cambridge University Press, to appear, 2007.

E. De Vito, A. Caponnetto, and L. Rosasco. Model selection for regularized least-squares algorithm in learning theory. *Foundations of Computational Mathematics*, 5:59–85, 2006.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, New York, 1997.

R. M. Dudley. *Uniform Central Limit Theorems*. Cambridge Studies in Advanced Mathematics 63, Cambridge University Press, 1999.

R. M. Dudley, E. Giné, and J. Zinn. Uniform and universal Glivenko-Cantelli Classes. *Journal of Theoretical Probability*, 4:485–510, 1991.

T. Evgeniou and M. Pontil. Regularized multi-task learning. In *Proceedings of 10th SIGKDD Conference on Knowledge Discovery and Data Mining*, 2004. http://www.cs.ucl.ac.uk/staff/M.Pontil/reading/mt-kdd.pdf

T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13:1–50, 2000.

V. Koltchinskii. Rademacher penalties and structural risk minimization. *IEEE Transactions on Information Theory*, 47:1902–1914, 2001.

V. Koltchinskii and V. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *The Annals of Statistics*, 30:1–50, 2002.

G. R. G. Lanckriet, N. Cristianini, P. L. Bartlett, P. L. El Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

M. Ledoux and M. Talagrand. *Probability in Banach Spaces: Isoperimetry and Processes*. Springer Press, New York, 1991.

J. Li, and A. Barron. Mixture density estimation. In *Advances in Neural Information Processing Systems* 12 (S. A. Solla, K. L. Todd, K.-R. Müller eds.), MIT Press, 2000.

P. Niyogi and F. Girosi. On the relationships between generalization error, hypothesis complexity and sample complexity for radial basis functions. *Neural Computation*, 8:819–842, 1996.

V. S. Pugachev, and I. N. Sinitsyn. *Lectures on Functional Analysis and Applications*. World Scientific, Singapore, 1999.

A. Rakhlin, D. Panchenko, and S. Mukherjee. Risk bounds for mixture density estimation. *ESAIM: Probability and Statistics*, 9:220–229, 2005.

B. Schölkopf, B. R. Herbrich, and A. J. Smola. A generalized representer theorem. In *Proceedings of the 14th Annual Conference on Computational Learning Theory, Lecture Notes in Artificial Intelligence*, 2111: 416–426, 2001.

J. Shawe-Taylor, P. L. Bartlett, S. C. Williamson, and M. Anthony. Structural risk minimization over data-dependent hierarchies. *IEEE Transactions on Information Theory*, 44:1926–1940, 1998.

S. Smale and D. X. Zhou. Estimating the approximation error in learning theory. *Analysis and Applications*, 1:17–41, 2003.

S. Smale and D. X. Zhou. Shannon sampling and function reconstruction from point values. *Bulletin of the American Mathematical Society*, 41:279–305, 2004.

S. Smale and D. X. Zhou. Shannon sampling II. Connections to learning theory. *Applied and Computational Harmonic Analysis*, 19:285–302, 2005.

E. M. Stein. *Singular Integrals and Differentiability Properties of Functions*. Princeton University Press, Princeton, New Jersey, 1970.

I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67–93, 2001.

I. Steinwart and C. Scovel. Fast rates for support vector machines. In *Proceedings of 18th Annual Conference on Learning Theory,* 2005.

A. B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *The Annals of Statistics*, 32:135–166, 2004.

V. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

A. W. Van der Vaart and J. A. Wellner. *Weak Convergence and Empirical Processes*. Springer-Verlag, 1996.

G. Wahba. *Spline Models for Observational Data*. SIAM, 1990.

Q. Wu, Y. Ying, and D. X. Zhou. Multi-kernel Regularized Classifiers. *Journal of Complexity*, forthcoming.

G. B. Ye and D. X. Zhou. Learning and approximation by Gaussians on Riemannian manifolds. *Advances in Computational Mathematics*, forthcoming.

K. Yosida. *Functional Analysis*. 6th edition, Springer-Verlag, 1980.

T. Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *The Annals of Statistics*, 32:56–85, 2004.

D. X. Zhou. The covering number in learning theory. *Journal of Complexity*, 18:739–767, 2002.

D. X. Zhou. Capacity of reproducing kernel spaces in learning theory. *IEEE Transactions on Information Theory*, 49:1743–1752, 2003.

D. X. Zhou and K. Jetter. Approximation with polynomial kernels and SVM classifiers. *Advances in Computational Mathematics*, 25:323–344, 2006.

Journal of Machine Learning Research 8 (2007) 277-290 Submitted 5/06; Revised 8/06; Published 2/07

Separating Models of Learning from Correlated and Uncorrelated Data

Ariel Elbaz ARIELBAZ@CS.COLUMBIA.EDU
Homin K. Lee HOMIN@CS.COLUMBIA.EDU
Rocco A. Servedio[*] ROCCO@CS.COLUMBIA.EDU
Andrew Wan ATW12@CS.COLUMBIA.EDU
Department of Computer Science
Columbia University
NY, NY 10027, USA

Editors: Peter Auer and Ron Meir

Abstract

We consider a natural framework of learning from correlated data, in which successive examples used for learning are generated according to a random walk over the space of possible examples. A recent paper by Bshouty et al. (2003) shows that the class of polynomial-size DNF formulas is efficiently learnable in this random walk model; this result suggests that the Random Walk model is more powerful than comparable standard models of learning from independent examples, in which similarly efficient DNF learning algorithms are not known. We give strong evidence that the Random Walk model is indeed more powerful than the standard model, by showing that if any cryptographic one-way function exists (a universally held belief in cryptography), then there is a class of functions that can be learned efficiently in the Random Walk setting but not in the standard setting where all examples are independent.

Keywords: random walks, uniform distribution learning, cryptographic hardness, correlated data, PAC learning

1. Introduction

It is a commonly held belief in machine learning that having access to correlated data—for example, having random data points that differ only slightly from each other—is advantageous for learning. Thus it is a natural research goal to rigorously validate this belief from the vantage point of the abilities and limitations of computationally efficient learning.

We study a natural model of learning from correlated data, by considering a framework in which the learning algorithm has access to successive examples that are generated by a *random walk*. We give strong evidence that learning is indeed easier, at least for some problems, in this framework of correlated examples than in the standard framework in which no correlations exist between successive examples.

*. Supported in part by NSF CAREER award CCF-0347282 and a Sloan Foundation Fellowship.

1.1 Background

In the well-known Probably Approximately Correct (PAC) learning model introduced by Valiant (1984), a learning algorithm is given access to a source $EX_{\mathcal{D}}(c)$ of labeled examples each of which is drawn *independently* from a fixed probability distribution $\mathcal{D}$ over the space of possible instances. The goal of the learning algorithm is to construct (with high probability) a high-accuracy hypothesis for the target concept c with respect to $\mathcal{D}$.

Aldous and Vazirani (1990) introduced and studied a variant of the PAC learning model in which successive examples are generated according to a Markov process, that is, by taking a random walk on an (exponentially large) graph. Subsequent work by Gamarnik (1999) extended this study to infinite Markov chains and gave bounds on the sample complexity required for learning in terms of the VC dimension and certain mixing properties of the underlying Markov chain. Neither Aldous and Vazirani (1990) nor Gamarnik (1999) considered computational issues for learning algorithms in the Random Walk framework.

In this paper we consider an elegant model of learning from Random Walk examples that is well suited for computational analyses. This model was introduced by Bartlett et al. (2002) and subsequently studied by Bshouty et al. (2003) and Roch (2006). In this framework (described in detail in Section 2), successive examples for the learning algorithm are produced sequentially according to an unbiased random walk on the Boolean hypercube $\{0,1\}^n$. The PAC goal of constructing a high-accuracy hypothesis for the target concept with high probability (where accuracy is measured with respect to the stationary distribution of the random walk, that is, the uniform distribution on $\{0,1\}^n$) is unchanged. This is a natural way of augmenting the model of uniform distribution PAC learning over the Boolean hypercube (which has been extensively studied, see, for example, Blum et al. 1994; Bshouty et al. 1999; Bshouty and Tamon 1996; Jackson 1997; Jackson et al. 2002; Kharitonov 1993; Linial et al. 1993; Verbeurgt 1990 and references therein) with the ability to exploit correlated data.

Bartlett *et al.* gave polynomial-time learning algorithms in this model for several concept classes including Boolean threshold functions in which each weight is either 0 or 1, parities of two monotone conjunctions over $x_1,\ldots,x_n$, and Disjunctive Normal Form (DNF) formulas with two terms. These learning algorithms are *proper*, meaning that in each case the learning algorithm constructs a hypothesis representation that belongs to the class being learned. Since proper learning algorithms were not known for these concept classes in the standard uniform distribution model, this gave the first evidence that having access to random walk examples rather than uniform independent examples might bestow a computational advantage.

More recently, Bshouty et al. (2003) gave a polynomial-time algorithm for learning the unrestricted class of all polynomial-size DNF formulas over $\{0,1\}^n$ in the Random Walk model. Since no comparable polynomial-time algorithms are known in the standard uniform distribution model—and their existence is a well-studied open question for which an affirmative answer would yield a \$1000 prize (see Blum, 2003)—this gives stronger evidence that the Random Walk model is strictly more powerful than the normal uniform distribution model. Thus, it is natural to now ask whether the superiority of random walk learning over uniform distribution learning can be established under some widely accepted hypothesis about efficient computation.

(Note that it is necessary to make some computational hardness assumption in order to separate these two learning models. It is well known and easy to see that if P=NP, for instance, then the concept class of all polynomial-size Boolean circuits would be efficiently learnable in both these models, as well as in far weaker models, by an algorithm that nondeterministically "guesses" a circuit and then checks consistency with a polynomial-size sample. Thus without some computational hardness assumption, essentially all considerations about the computational complexity of learning would become trivial.)

1.2 Our Results

In this work we give a separation, under a generic cryptographic hardness assumption, between the Random Walk model and the uniform distribution model. Our main result is a proof of the following theorem:

Theorem 1 *If any cryptographic one-way function exists, then there is a concept class over* $\{0,1\}^n$ *that is PAC learnable in poly*(n) *time in the Random Walk model but is not PAC learnable in poly*(n) *time in the standard uniform distribution model.*

We emphasize that the separation established by Theorem 1 is computational rather than information-theoretic. It will be evident from our construction that the concept class of Theorem 1 has poly(n) VC dimension, and thus the class can be learned using poly(n) many *examples* even in the distribution-independent PAC learning model; the difficulty is in obtaining a *polynomial-time* algorithm.

We remind the reader that while the existence of any one-way function is a stronger assumption than the assumption that P≠NP (since at this point it is conceivable that P≠NP but one-way functions do not exist), it is an almost universally accepted assumption in cryptography and complexity theory. (In particular, the existence of one-way functions is the weakest of the many assumptions on which the entire field of public-key cryptography is predicated.) We also remind the reader that all known representation-independent computational hardness results in learning theory (where any efficiently evaluatable hypothesis representation is allowed for the learning algorithm, as is the case in Theorem 1 above) rely on cryptographic hardness assumptions rather than complexity-theoretic assumptions such as P≠NP.

The rest of the paper is structured as follows: Section 2 gives necessary definitions and background from cryptography and the basics of our random walk model. Section 3 gives a partial separation, and in Section 4 we show how the construction from Section 3 can be used to achieve a total separation and prove Theorem 1.

2. Preliminaries

We denote by $[n]$ the set $\{1,\dots,n\}$. For an n-bit string $r \in \{0,1\}^n$ and an index $i \in [n]$, the i-th bit of r is denoted $r[i]$. We write $\mathcal{U}$ to denote the uniform distribution on $\{0,1\}^n$.

2.1 Learning Models

Recall that a concept class $\mathcal{C} = \cup_{n\in\mathbb{N}}\mathcal{C}_n$ is a collection of Boolean functions where each $f \in \mathcal{C}_n$ maps $\{0,1\}^n \to \{0,1\}$. A *uniform example oracle* for f is an oracle $EX_U(f)$ which takes no inputs and, when invoked, outputs a pair $\langle x, f(x)\rangle$ where x is drawn uniformly and independently from $\{0,1\}^n$ at each invocation.

Definition 2 (PAC learning) *A concept class* $\mathcal{C}$ *is* uniform distribution PAC-learnable *if there is an algorithm A with the following property: for any n, any target concept* $f \in \mathcal{C}_n$, *and any* $\varepsilon, \delta > 0$, *if A* *is given access to oracle $EX_U(f)$ then A runs for poly$(n, \frac{1}{\varepsilon}, \frac{1}{\delta})$ time steps and with probability* $1-\delta$ *outputs a Boolean circuit h such that* $\Pr_{x \in \mathcal{U}}[h(x) \neq c(x)] \leq \varepsilon$.

In the (uniform) Random Walk model studied in Bartlett et al. (2002) and Bshouty et al. (2003), a *random walk* oracle is an oracle $EX_{RW}(f)$ which, at its first invocation, outputs an example $\langle x, f(x)\rangle$ where x is drawn uniformly at random from $\{0,1\}^n$. Subsequent calls to $EX_{RW}(f)$ yield examples generated according to a uniform random walk on the hypercube $\{0,1\}^n$. That is, if x is the i-th example, the $i+1$-st example is x', where x' is chosen by uniformly selecting one of the n bits of x and flipping it.

Definition 3 (PAC learning in the Random Walk model) *A concept class $\mathcal{C}$ is said to be* PAC-learnable in the Random Walk model *if there is an algorithm A that satisfies Definition 2 above but with $EX_{RW}(f)$ in place of $EX_U(f)$.*

As in Bshouty et al. (2003), it is convenient for us to work with a slight variant of the Random Walk oracle which is of equivalent power; we call this the *Updating Random Walk* oracle and denote it by $EX_{URW}(f)$. If the last example generated by $EX_{URW}(f)$ was $x \in \{0,1\}^n$, the Updating Random Walk oracle chooses a uniform index $i \in [n]$, but instead of flipping the bit $x[i]$ it replaces $x[i]$ with a uniform random bit from $\{0,1\}$ (i.e., it flips the bit with probability $1/2$ and leaves x unchanged with probability $1/2$) to obtain the new example x'. We say that such a step *updates* the i-th bit position.

It is easy to simulate an Updating Random Walk oracle using a Random Walk oracle by, at each time step, flipping a fair coin and with probability $1/2$ returning the previous example and with probability $1/2$ requesting a new example. Similarly, it is easy to simulate a Random Walk oracle using an Updating Random Walk oracle by invoking the Updating Random Walk oracle until it flips a bit (the probability that this takes more than k invocations is at most 2^{-k}). Thus any concept class that is efficiently learnable from one oracle is also efficiently learnable from the other. We introduce the Updating Random Walk oracle because it is easy to see (and well known) that the updating random walk on the hypercube mixes rapidly. More precisely, we have the following fact which will be useful later:

Fact 4 *Let $\langle x, f(x)\rangle$ be a labeled example that is obtained from $EX_{URW}(f)$, and let $\langle y, f(y)\rangle$ be the labeled example that $EX_{URW}(f)$ outputs $n \ln \frac{n}{\delta}$ draws later. Then with probability at least $1-\delta$, the two strings x, y are uniformly and independently distributed over $\{0,1\}^n$.*

Proof Since it is clear that x and y are each uniformly distributed, the only thing to check for Fact 4 is independence. This follows since y will be independent of x if and only if all n bit positions are updated in the $n \ln \frac{n}{\delta}$ draws between x and y. For each draw, the probability that a particular bit is not updated is $(1-\frac{1}{n})$. Thus after $n \ln \frac{n}{\delta}$ draws, the probability that any bit of r has not been updated is at most $n(1-\frac{1}{n})^{n \ln \frac{n}{\delta}} \leq \delta$. This yields the fact. ■

Note that Fact 4 implies that any concept class $\mathcal{C}$ that is uniform distribution PAC-learnable is also PAC-learnable in the Random Walk model, since we can obtain independent uniform random examples in the Random Walk model with essentially just a $\Theta(n \log n)$ slowdown.

2.2 Background from Cryptography

We write $\mathcal{R}_n$ to denote the set of all 2^{2^n} Boolean functions from $\{0,1\}^n$ to $\{0,1\}$. We refer to a function f chosen uniformly at random from $\mathcal{R}_n$ as a *truly random function*. We write D^f to denote a probabilistic polynomial-time (p.p.t.) algorithm D with black-box oracle access to the function f.

Informally, a one-way function is a function $f : \{0,1\}^n \to \{0,1\}^n$ that is computable by a $\text{poly}(n)$ time algorithm but is hard to invert in the sense that no $\text{poly}(n)$-time algorithm can successfully compute f^{-1} on a nonnegligible fraction of outputs of f. (See Goldreich 2001 for a detailed definition and discussion of one-way functions.) In a celebrated result, Håstad et al. (1999) showed that if any one-way function exists, then *pseudorandom function families* must exist as well.

Definition 5 *A* pseudorandom function family *(Goldreich et al., 1986) is a collection of functions* $\{f_s : \{0,1\}^{|s|} \to \{0,1\}\}_{s\in\{0,1\}^*}$ *with the following two properties:*

1. (efficient evaluation) *there is a deterministic algorithm which, given an n-bit seed s and an n-bit input x, runs in time poly(n) and outputs* $f_s(x)$;

2. (pseudorandomness) *for all polynomials Q, all p.p.t. oracle algorithms D, and all sufficiently large n, we have that*

$$\left| \Pr_{f\in\mathcal{R}_n}[D^f(1^n) \texttt{ outputs } 1] - \Pr_{s\in\{0,1\}^n}[D^{f_s}(1^n) \texttt{ outputs } 1] \right| < \frac{1}{Q(n)}.$$

The argument 1^n indicates that the "distinguisher" algorithm D must run in $\text{poly}(n)$ time steps since its input is of length n. Intuitively, condition (2) above states that a pseudorandom function cannot be distinguished from a truly random function by any polynomial-time algorithm that has black-box access to the pseudorandom function with an inverse polynomial advantage over random guessing.

3. A Partial Separation

We will first show the difficulties presented with an obvious separation, and then show a partial separation that will be the building block of the full separation.

3.1 A First Attempt

It is clear that in the Random Walk model a learning algorithm will get many pairs of examples that are adjacent vertices of the Hamming cube $\{0,1\}^n$, whereas this will not be the case for a learner in the standard uniform distribution model (with high probability, a set of $\text{poly}(n)$ many independent uniform examples from $\{0,1\}^n$ will contain no pair of examples that have Hamming distance less than $n/2 - O(\sqrt{n\log n})$). Thus, in attempting to separate the random walk model from the standard uniform distribution model, it is natural to try to construct a concept class using pseudorandom functions f_s but altered in such a way that seeing the value of the function on adjacent inputs gives away information about the seed s.

One natural approach is the following: given a pseudorandom function family $\{f_s : \{0,1\}^k \to \{0,1\}\}_{s\in\{0,1\}^k}$, one could define a concept class of functions $\{f'_s : \{0,1\}^k \times \{0,1\}^{\log k} \times \{0,1\} \to$

$\{0,1\}\}_{s\in\{0,1\}^k}$ as follows (so $n = k + \log k + 1$):

$$f'_s(x,i,b) = \begin{cases} f_s(x) & \text{if } b = 0, \\ f_s(x) \oplus s[i] & \text{if } b = 1, \end{cases}$$

where x is a k-bit string, i is a $(\log k)$-bit string encoding an integer between 1 and k, and b is a single bit. A learning algorithm in the Random Walk model will be able to obtain all bits $s[1], \ldots, s[k]$ of the seed s (by waiting for pairs of successive examples $(x,i,b),(x,i,1-b)$ in which the final bit b flips for all k possible values of i), and will thus be able to exactly identify the target concept. However, even though a standard uniform distribution learner will not obtain any pair of inputs that differ only in the final bit b, it is not clear how to show that no algorithm in the standard uniform distribution model can learn the concept class to high accuracy. Such a proof would require one to show that any polynomial-time uniform distribution learning algorithm could be used to "break" the pseudorandom function family $\{f_s\}$, and this seems difficult to do. (Intuitively, this difficulty arises because the $b = 1$ case of the definition of f'_s "mixes" bits of the seed with the output of the pseudorandom function, and because of this it is not clear how to simulate f'_s given black-box access to f_s for an unknown seed s.) Thus, we consider alternate constructions.

3.2 A Partial Separation

In this section we describe a concept class and prove that it has the following two properties: (1) A randomly chosen concept from the class is indistinguishable from a truly random function to any polynomial-time algorithm which has an $EX_U(\cdot)$ oracle for the concept (and thus no such algorithm can learn to accuracy $\varepsilon = \frac{1}{2} - \frac{1}{\mathrm{poly}(n)}$); (2) However, a Random Walk algorithm with access to $EX_{RW}(\cdot)$ can learn any concept in the class to accuracy $\frac{3}{4}$. In the next section we will extend this construction to fully separate the Random Walk model from the standard uniform model and thus prove Theorem 1.

Our construction uses ideas from Section 3.1; as in the construction proposed there, the concepts in our class will reveal information about the seed of a pseudorandom function to learning algorithms that can obtain pairs of points with only the last bit flipped. However, each concept in the class will now be defined by *two* pseudorandom functions rather than one; this will enable us to prove that the class is indeed hard to learn in the uniform distribution model (but will also prevent a Random Walk learning algorithm from learning to accuracy better than 3/4).

Let $\mathcal{F}$ be a family of pseudorandom functions $\{f_r : \{0,1\}^k \to \{0,1\}\}_{r\in\{0,1\}^k}$. We construct a concept class $\mathcal{G} = \{g_{r,s} : r,s \in \{0,1\}^k\}$, where $g_{r,s}$ takes an n-bit input that we split into four parts for convenience. As before, the first k bits x give the "actual" input to the function, while the other parts determine the mode of function that will be applied.

$$g_{r,s}(x,i,b,y) = \begin{cases} f_s(x) & \text{if } y = 0, b = 0, \\ f_s(x) \oplus r[i] & \text{if } y = 0, b = 1, \\ f_r(x) & \text{if } y = 1. \end{cases}$$

Here b and y are one bit and i is $\log k$ bits to indicate which bit of the seed r is exposed. Thus half of the inputs to $g_{r,s}$ are labeled according to f_r, and the other half are labeled according to either f_s or $f_s \oplus r[i]$ depending on the value of b.

The following lemma establishes that $\mathcal{G}$ is not efficiently PAC-learnable under the uniform distribution, by showing that a random function from $\mathcal{G}$ is indistinguishable from a truly random function to any algorithm which only has $EX_U(\cdot)$ access to the target concept. (A standard argument shows that an efficient PAC learning algorithm can be used to obtain an efficient distinguisher simply by running the learning algorithm and using its hypothesis to predict a fresh random example. Such an approach must succeed with high probability for any function from the concept class by virtue of the PAC criterion, but no algorithm that has seen only poly(n) many examples of a truly random function can predict its outputs on fresh examples with probability nonnegligibly greater than $\frac{1}{2}$.)

Lemma 6 *Fix any p.p.t. algorithm A. Let $g_{r,s} : \{0,1\}^n \to \{0,1\}$ be a function from $\mathcal{G}$ chosen by selecting r and s uniformly at random from $\{0,1\}^k$, where k satisfies $n = k + \log k + 2$. Let f be a truly random function. Then for any $\varepsilon = \Omega(\frac{1}{\text{poly}(n)})$, algorithm A cannot distinguish between having oracle access to $EX_U(g_{r,s})$ versus oracle access to $EX_U(f)$ with success probability greater than $\frac{1}{2} + \varepsilon$.*

Proof The proof is by a hybrid argument. We will construct two intermediate functions, h_r and h'_r. We will show that $EX_U(g_{r,s})$ is indistinguishable from $EX_U(h_r)$, $EX_U(h_r)$ from $EX_U(h'_r)$, and $EX_U(h'_r)$ from $EX_U(f)$. It will then follow that $EX_U(g_{r,s})$ is indistinguishable from $EX_U(f)$.

Consider the function

$$h_r(x,i,b,y) = \begin{cases} f(x) & \text{if } y = 0, b = 0, \\ f(x) \oplus r[i] & \text{if } y = 0, b = 1, \\ f_r(x) & \text{if } y = 1. \end{cases} \tag{1}$$

Here we have simply replaced f_s with a truly random function. We claim that no p.p.t. algorithm can distinguish oracle access to $EX_U(g_{r,s})$ from oracle access to $EX_U(h_r)$; for if such a distinguisher D existed, we could use it to obtain an algorithm D' to distinguish a randomly chosen $f_s \in \mathcal{F}$ from a truly random function in the following way. D' picks r at random from $\{0,1\}^k$ and runs D, answering D's queries to its oracle by choosing i, b and y at random, querying its own oracle to receive a bit q, and outputting q when both y and b are 0, $q \oplus r[i]$ when $y = 0$ and $b = 1$, and $f_r(x)$ when $y = 1$. It is easy to see that if D''s oracle is for a truly random function $f \in \mathcal{R}$ then this process perfectly simulates access to $EX_U(h_r)$, and if D''s oracle is for a randomly chosen $f_s \in \mathcal{F}$ then this process perfectly simulates access to $EX_U(g_{r,s})$ for r, s chosen uniformly at random.

We now consider the intermediate function

$$h'_r(x,i,b,y) = \begin{cases} f(x) & \text{if } y = 0, \\ f_r(x) & \text{if } y = 1, \end{cases}$$

and argue that no algorithm that makes only poly(n) many oracle calls can distinguish oracle access to $EX_U(h_r)$ from access to $EX_U(h'_r)$ with non-negligible success probability. When $y = 1$ or both $y = 0$ and $b = 0$, both h_r and h'_r will have the same output. Otherwise, if $y = 0$ and $b = 1$ we have that $h_r(x,i,b,y) = f(x) \oplus r_i$ whereas $h'_r(x,i,b,y) = f(x)$. Now, it is easy to see that an algorithm with *black-box query access* to h_r can easily distinguish h_r from h'_r (simply because flipping the penultimate bit b will always cause the value of h_r to flip but will only cause the value of h'_r to flip half of the time). But for an algorithm that only has oracle access to $EX_U(\cdot)$, conditioned on never receiving the same string x twice (a condition that fails to hold only with inverse exponential probability for any algorithm that makes poly(n) many oracle calls), it is easy to see that whether the oracle is for h_r or h'_r, each output value that the algorithm sees on inputs with $y = 0$ and $b = 1$ will be a fresh independent uniform random bit. (This is simply because a random function f can be viewed as tossing a coin to determine its output on each new input value, so no matter what $r[i]$ is, XORing it with $f(x)$ yields a fresh independent uniform random bit.)

Finally, it follows from the definition of pseudorandomness that no p.p.t. algorithm can distinguish oracle access to $EX_U(h'_r)$ from access to $EX_U(f)$. We have thus shown that $EX_U(g_{r,s})$ is indistinguishable from $EX_U(h_r)$, $EX_U(h_r)$ from $EX_U(h'_r)$, and $EX_U(h'_r)$ from $EX_U(f)$. It follows that $EX_U(g_{r,s})$ is indistinguishable from $EX_U(f)$, and the proof is complete. ■

We now show that $g_{r,s}$ is learnable to accuracy $\frac{3}{4}$ in the Random Walk model. The basic idea is that each time bit b is flipped on an example (x,i,b,y) with $y = 0$, the value of $r[i]$ is revealed, and this happens for all i within poly(n) many steps.

Lemma 7 *There is an algorithm A with the following property: for any $\delta > 0$ and any concept $g_{r,s} \in \mathcal{G}$, if A is given access to a Random Walk oracle $EX_{RW}(g_{r,s})$ then A runs in time poly$(n, \log(1/\delta))$ and with probability at least $1-\delta$, algorithm A outputs an efficiently computable hypothesis h such that $\Pr_{\mathcal{U}}[h(x) \neq g_{r,s}(x)] \leq \frac{1}{4}$.*

Proof As described in Section 2, for convenience in this proof we will assume that we have an Updating Random Walk oracle $EX_{URW}(g_{r,s})$.

We give an algorithm that, with probability $1-\delta$, learns all the bits of r. Once the learner has obtained r she outputs the following (randomized) hypothesis h:

$$h(x,i,b,y) = \begin{cases} \$ & \text{if } y = 0, \\ f_r(x) & \text{if } y = 1, \end{cases}$$

where \$ denotes a random coin toss at each invocation. Note that h incurs zero error relative to $g_{r,s}$ on inputs that have $y = 1$, and has error rate exactly $\frac{1}{2}$ on inputs that have $y = 0$. Thus the overall error rate of h is exactly $\frac{1}{4}$.

We now show that with probability $1-\delta$ (over the random examples received from $EX_{URW}(g_{r,s})$) the learner can obtain all of r after receiving $T = O(n^2 k \cdot \log^2(n/\delta))$ many examples from $EX_{URW}(g_{r,s})$. The learner does this by looking at pairs of successive examples; we show (Fact 10 below) that after seeing $t = O(nk \cdot \log(k/\delta))$ pairs, each of which is independent from all other pairs, we obtain all of r with probability at least $1-\frac{\delta}{2}$. To get t independent pairs of successive examples, we look at blocks of $t' = O(n \log(tn/\delta))$ many consecutive examples, and use only the first two examples from each such block. By Fact 4 we have that for a given pair of consecutive blocks, with probability at least $1-\frac{\delta}{2t}$ the first example from the second block is random even given the pair of examples from the first block. A union bound over the t blocks gives total failure probability at most $\frac{\delta}{2}$ for independence, and thus an overall failure probability of at most δ.

We have the following simple facts:

Fact 8 *If the learner receives two consecutive examples $w = (x,i,0,0), w' = (x,i,1,0)$ and the corresponding labels $g_{r,s}(w), g_{r,s}(w')$, then the learner can obtain the bit $r[i]$.*

Fact 9 *For any $j \in [k]$, given a pair of consecutive examples from $EX_{URW}(g_{r,s})$, a learning algorithm can obtain the value of $r[j]$ from this pair with probability at least $\frac{1}{4kn}$.*

Proof By Fact 8, if the first example is $w = (x, i, b, y)$ with $i = j, y = 0$ and the following example differs in the value of b, then the learner obtains $r[j]$. The first example (like every example from $EX_{URW}(g_{r,s})$) is uniformly distributed and thus has $i = j, y = 0$ with probability $\frac{1}{2k}$. The probability that the next example from $EX_{URW}(g_{r,s})$ flips the value of b is $\frac{1}{2n}$. ■

Fact 10 *After receiving $t = 4kn \cdot \log(k/\delta')$ independent pairs of consecutive examples as described above, the learner can obtain all k bits of r with probability at least $1 - \delta'$.*

Proof For any $j \in [k]$, the probability that $r[j]$ is not obtained from a given pair of consecutive examples is at most $(1 - \frac{1}{4kn})$. Thus after seeing t independent pairs of consecutive examples, the probability that any bit of r is not obtained is at most $k(1 - \frac{1}{4kn})^t$. This yields the fact. ■

Thus the total number of calls to $EX_{URW}(g_{r,s})$ that are required is:

$$T = t \cdot t' = O(nk \log(k/\delta)) \cdot O(n \log(tn/\delta)) = O(n^2 k \log^2(n/\delta)).$$

Since $k = O(n)$, Lemma 7 is proved. ■

4. A Full Separation

We would like to have a concept class for which a Random Walk learner can output an ϵ-accurate hypothesis for any $\epsilon > 0$. The drawback of our construction in Section 3.2 is that a Random Walk learning algorithm can only achieve a particular fixed error rate $\epsilon = \frac{1}{4}$. Intuitively, a Random Walk learner cannot achieve accuracy better than $\frac{3}{4}$ because on half of the inputs the concept's value is essentially determined by a pseudorandom function whose seed the Random Walk learner cannot discover. It is not difficult to see that for any given $\epsilon = \frac{1}{\text{poly}(n)}$, by altering the parameters of the construction we could obtain a concept class that a Random Walk algorithm can learn to accuracy $1 - \epsilon$ (and which would still be unlearnable for a standard uniform distribution algorithm). However, this would give us a different concept class for each ϵ, whereas what we require is a single concept class that can be learned to accuracy ϵ for each $\epsilon > 0$.

In this section we present a new concept class $\mathcal{G}'$ and show that it achieves this goal. The idea is to string together many copies of our function from Section 3.2 in a particular way. Instead of depending on two seeds r, s, a concept in $\mathcal{G}'$ is defined using k seeds $r_1, \ldots, r_k$ and $k - 1$ subfunctions $g_{r_1,r_2}, g_{r_2,r_3}, \ldots, g_{r_{k-1},r_k}$. These subfunctions are combined in a way that lets the learner learn more and more of the seeds $r_1, r_2, \ldots$, and thus learn to higher and higher accuracy, as she receives more and more examples.

4.1 The Concept Class $\mathcal{G}'$

We now describe $\mathcal{G}'$ in detail. Each concept in $\mathcal{G}'$ is defined by k seeds $r_1, \dots, r_k$, each of length k. The concept $g'_{r_1,\dots,r_k}$ is defined by

$$g'_{r_1,\dots,r_k}(x,i,b,y,z) = \begin{cases} g_{r_{\alpha(z)},r_{\alpha(z)+1}}(x,i,b,y) & \text{if } \alpha(z) \in \{1,\dots,k-1\}, \\ f_{r_k}(x) & \text{if } \alpha(z) = k. \end{cases}$$

As in the previous section x is a k-bit string, i is a $\log k$-bit string, and b and y are single bits. The new input z is a $(k-1)$-bit string, and the value $\alpha(z) \in [k]$ is defined as the index of the leftmost bit in z that is 1 (for example if $z = 0010010111$ then $\alpha(z) = 3$); if $z = 0^{k-1}$ then $\alpha(z)$ is defined to be k. By this design, the subfunction $g_{r_j,r_{j+1}}$ will be used on a $1/2^j$ fraction of the inputs to g'. Note that g' maps $\{0,1\}^n$ to $\{0,1\}$ where $n = 2k + \log k + 1$.

4.2 Uniform Distribution Algorithms Cannot Learn $\mathcal{G}'$

We first show that $\mathcal{G}'$ is not efficiently PAC-learnable under the uniform distribution. This is implied by the following lemma:

Lemma 11 *Fix any p.p.t. algorithm A. Let $g'_{r_1,\dots,r_k} : \{0,1\}^n \to \{0,1\}$ be a function from $\mathcal{G}'$ chosen by selecting $r_1, \dots, r_k$ uniformly at random from $\{0,1\}^k$, where k satisfies $n = 2k + \log k + 1$. Let f be a truly random function. Then for any $\varepsilon = \Omega(\frac{1}{\mathrm{poly}(n)})$, algorithm A cannot distinguish between having access to $EX_U(g'_{r_1,\dots,r_k})$ versus access to $EX_U(f)$ with success probability greater than $\frac{1}{2} + \varepsilon$.*

Proof Again we use a hybrid argument. We define the concept classes $\mathcal{H}(\ell) = \{h_{r_1,\dots,r_\ell;f} : r_1, \dots, r_\ell \in \{0,1\}^k, f \in \mathcal{R}_k\}$ for $2 \le \ell \le k$. Each function $h_{r_1,\dots,r_\ell;f}$ takes the same n-bit input (x,i,b,y,z) as $g'_{r_1,\dots,r_k}$. The function $h_{r_1,\dots,r_\ell;f}$ is defined as follows:

$$h_{r_1,\dots,r_\ell;f}(x,i,b,y,z) = \begin{cases} g_{r_{\alpha(z)},r_{\alpha(z)+1}}(x,i,b,y) & \text{if } \alpha(z) < \ell, \\ f(x) & \text{otherwise.} \end{cases}$$

Here as before, the value $\alpha(z) \in [k]$ denotes the index of the leftmost bit of z that is one (and we have $\alpha(z) = k$ if $z = 0^{k-1}$).

We will consider functions that are chosen uniformly at random from $\mathcal{H}(\ell)$, that is, $r_1, \dots, r_\ell$ are chosen randomly from $\{0,1\}^k$ and f is a truly random function from $\mathcal{R}_k$. Using Lemma 6, it is easy to see that for a distinguisher that is given only oracle access to $EX_U(\cdot)$, a random function from $\mathcal{H}(2)$ is indistinguishable from a truly random function from $\mathcal{R}_n$. We will now show that, for $2 \le \ell < k$, if a random function from $\mathcal{H}(\ell)$ is indistinguishable from a truly random function then the same is true for $\mathcal{H}(\ell+1)$. This will then imply that a random function from $\mathcal{H}(k)$ is indistinguishable from a truly random function.

Let $h_{r_1,\dots,r_{\ell+1};f}$ be taken randomly from $\mathcal{H}(\ell+1)$ and f be a truly random function from $\mathcal{R}_n$. Suppose we had a distinguisher D that distinguishes between a random function from $\mathcal{H}(\ell+1)$ and a truly random function from $\mathcal{R}_n$ with success probability $\frac{1}{2} + \varepsilon$, where $\varepsilon = \Omega(\frac{1}{\mathrm{poly}(n)})$. Then we can use D to obtain an algorithm D' for distinguishing a randomly chosen $f_s \in \mathcal{F}$ from a randomly chosen function $f \in \mathcal{R}_k$ in the following way. D' first picks strings $r_1, \dots, r_\ell$ at random from $\{0,1\}^k$. D' then runs D, simulating its oracle in the following way. At each invocation, D' draws a random (x,i,b,y,z) and behaves as follows:

- If $\alpha(z) < \ell$, then D' outputs $\langle (x,i,b,y,z), g_{r_{\alpha(z)},r_{\alpha(z)+1}}(x,i,b,y)\rangle$.
- If $\alpha(z) = \ell$, then D' calls its oracle to obtain $\langle x', \beta\rangle$. If $y = b = 0$ then D' outputs $\langle (x',i,b,y,z), \beta\rangle$. If $y = 0$ but $b = 1$ then D' outputs $\langle (x',i,b,y,z), \beta \oplus r_\ell[i]\rangle$. If $y = 1$ then D' outputs $\langle (x',i,b,y,z), f_{r_\ell}(x)\rangle$.
- If $\alpha(z) > \ell$, D' outputs the labeled example $\langle (x,i,b,y,z), r(x)\rangle$ where $r(x)$ is a fresh random bit for each x. (The pairs $(x, r(x))$ are stored, and if any k-bit string x is drawn twice—which is exponentially unlikely in a sequence of poly(n) many draws—D' uses the same bit $r(x)$ as before.)

It is straightforward to check that if D''s oracle is $EX_U(f_s)$ for a random $f_s \in \mathcal{F}$, then D' simulates an oracle $EX_U(h_{r_1,\dots,r_{\ell+1};f})$ for D, where $h_{r_1,\dots,r_{\ell+1};f}$ is drawn uniformly from $\mathcal{H}(\ell+1)$. On the other hand, we claim that if D''s oracle is $EX_U(f)$ for a random $f \in \mathcal{R}_k$, then D' simulates an oracle that is indistinguishable from $EX_U(h_{r_1,\dots,r_\ell;f})$ for D, where $h_{r_1,\dots,r_\ell;f}$ is drawn uniformly from $\mathcal{H}(\ell)$. Clearly the oracle D' simulates is identical to $EX_U(h_{r_1,\dots,r_\ell;f})$ for $\alpha(z) \neq \ell$. For $\alpha(z) = \ell$, D' simulates the function h_{r_ℓ} as in Equation 1 in the proof of Lemma 6, which is indistinguishable from a truly random function as proved in the lemma.

Thus the success probability of the distinguisher D' is the same as the probability that D succeeds in distinguishing $\mathcal{H}(\ell+1)$ from $\mathcal{H}(\ell)$. Recall that $\mathcal{H}(\ell)$ is indistinguishable from a truly random function, and that D succeeds in distinguishing $\mathcal{H}(\ell+1)$ from a truly random function with probability at least $\frac{1}{2}+\varepsilon$ by assumption. This implies that D' succeeds in distinguishing a randomly chosen $f_s \in \mathcal{F}$ from a randomly chosen function $f \in \mathcal{R}_k$ with probability at least $\frac{1}{2}+\varepsilon-\frac{1}{\omega(\text{poly}(n))}$, but this contradicts the pseudorandomness of $\mathcal{F}$.

Finally, we claim that for any p.p.t. algorithm, having oracle access to a random function from $\mathcal{H}(k)$ is indistinguishable from having oracle access to a random function from $\mathcal{G}'$. To see this, note that the functions $h_{r_1,\dots,r_\ell;f}$ and $g'_{r_1,\dots,r_\ell}$ differ only on inputs (x,i,b,y,z) that have $\alpha(z) = k$, that is, $z = 0^{k-1}$ (on such inputs the function $g_{r_1,\dots,r_\ell}$ will output $f_{r_k}(x)$ whereas $h_{r_1,\dots,r_\ell;f}$ will output $f(x)$). But such inputs are only a $\frac{1}{2^{\Omega(n)}}$ fraction of all possible inputs, so with overwhelmingly high probability a p.p.t. algorithm will never receive such an example. ■

4.3 Random Walk Algorithms Can Learn $\mathcal{G}'$

The following lemma completes the proof of our main result, Theorem 1.

Lemma 12 *There is an algorithm B with the following property: for any $\varepsilon,\delta > 0$, and any concept $g_{r_1,\dots,r_k} \in \mathcal{G}'$, if B is given access to a Random Walk oracle $EX_{RW}(g_{r_1,\dots,r_k})$, then B runs in time poly$(n, \log(1/\delta), 1/\varepsilon)$ and can with probability at least $1-\delta$ output a hypothesis h such that* $\Pr_U[h(x) \neq g_{r_1,\dots,r_k}(x)] \leq \varepsilon$.

Proof The proof is similar to that of Lemma 7. Again, for convenience we will assume that we have an Updating Random Walk oracle $EX_{URW}(g_{r_1,\dots,r_k})$. Recall from Lemma 7 that there is an algorithm A that can obtain the string r_j with probability at least $1-\delta'$ given $t' = O(nk \cdot \log(n/\delta'))$ independent pairs of successive random walk examples

$$\left(\langle w, g_{r_j,r_{j+1}}(w)\rangle, \langle w', g_{r_j,r_{j+1}}(w')\rangle\right).$$

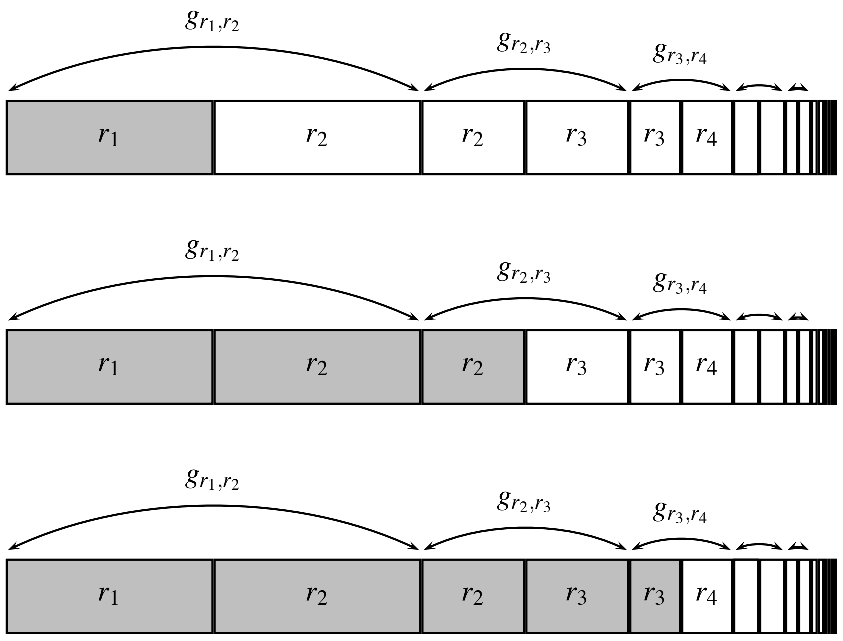


Figure 1: Stages $1, 2$ and 3 of Algorithm B. Each row represents the output values of $g'_{r_1,\ldots,r_k}$. After stage j the algorithm "knows" $r_1,\ldots,r_j$ and can achieve perfect accuracy on the shaded region.

Algorithm B works in a sequence of v stages. In stage j, the algorithm simply tries to obtain t' independent example pairs for $g_{r_j,r_{j+1}}$ and then uses Algorithm A. Assuming the algorithm succeeds in each stage, after stage v algorithm B has obtained $r_1,\ldots,r_v$. It follows directly from the definition of $\mathcal{G}'$ that given $r_1,\ldots,r_v$, Algorithm B can construct a hypothesis that has error at most $\frac{3}{2^{v+2}}$ (see Figure 1) so we may take $v = \log\frac{1}{\varepsilon}+1$ to obtain error at most ε. (Note that this implicitly assumes that $\log\frac{1}{\varepsilon}+1$ is at most k; we deal with the case $\log\frac{1}{\varepsilon}+1 > k$ at the end of the proof.)

If the learner fails to obtain $r_1,\ldots,r_v$, then either:

1. Independence was not achieved between every pair of examples;

2. Algorithm B fails to acquire t' pairs of examples for $g_{r_j,r_{j+1}}$ in some stage j; or

3. Algorithm B acquires t' pairs of examples for $g_{r_j,r_{j+1}}$ but Algorithm A fails to obtain r_j in some stage j.

We choose the total number of examples so that each of these probabilities is bounded by $\delta/3$ to achieve an overall failure probability of at most δ.

As will be clear from the analysis of cases (2) and (3) below, in total Algorithm B will use $4\cdot 2^{v+1}t'$ pairs of examples in stages 1 through v, where t' will be bounded later. Each pair of examples is obtained by using the first two examples from a block of $s = O(n\log(v\cdot 2^{v+1}t'n/\delta))$

many consecutive examples from the Updating Random Walk oracle. With this choice of s, the same argument as in the proof of Lemma 7 shows that the total failure probability for independence is at most $\frac{\delta}{3}$.

We bound (2) assuming full independence between all pairs of examples. In stage j, Algorithm B uses $4 \cdot 2^j t'$ pairs of examples. Observe that each pair of examples has both examples from $g_{r_j,r_{j+1}}$ with probability at least $2^{-(j+1)}$. By a Chernoff bound, the probability that less than t' of the example pairs in stage j are from $g_{r_j,r_{j+1}}$ is at most $e^{-\frac{t'}{8}}$. Thus the overall probability of failure from condition (2) is at most $ve^{-\frac{t'}{8}}$ which is at most $\delta/3$ for $t' \geq \ln(3v/\delta)$.

We bound (3) assuming full independence between all pairs of examples as well. In stage j, we know by Fact 10 that after seeing $t' = O(nk\log(3vk/\delta))$ pairs of examples for $g_{r_j,r_{j+1}}$, the probability of failing to obtain r_j is at most $\delta/3v$. Hence the overall failure probability from condition (3) is at most $\frac{\delta}{3}$.

We thus may take $t' = O(nk\log(3vk/\delta))$ and achieve an overall failure probability of δ for obtaining $r_1,\ldots,r_v$. It follows that the overall number of examples required from the Updating Random Walk oracle is $\mathrm{poly}(2^v, n, \log\frac{1}{\delta}) = \mathrm{poly}(n, \frac{1}{\varepsilon}, \log\frac{1}{\delta})$, which is what we required.

Finally, we observe that if $\log\frac{1}{\varepsilon} + 1 > k$, since $k = \frac{n}{2} - O(\log n)$ a $\mathrm{poly}(\frac{1}{\varepsilon})$-time algorithm may run for, say, 2^{2n} time steps and thus build an explicit truth table for the function. Such a table can be used to exactly identify each seed $r_1,\ldots,r_k$ and output an exact representation of the target concept. ■

Acknowledgments

We warmly thank Tal Malkin for helpful discussions and the anonymous referees for useful suggestions that improved the presentation and readability of the paper.

References

D. Aldous and U. Vazirani. A Markovian extension of Valiant's learning model. In *Proceedings of the Thirty-First Symposium on Foundations of Computer Science*, pages 392–396, 1990.

P. Bartlett, P. Fischer, and K.U. Höffgen. Exploiting random walks for learning. *Information and Computation*, 176(2):121–135, 2002.

A. Blum. Learning a function of *r* relevant variables (open problem). In *Proc. 16th Annual COLT*, pages 731–733, 2003.

A. Blum, M. Furst, J. Jackson, M. Kearns, Y. Mansour, and S. Rudich. Weakly learning DNF and characterizing statistical query learning using Fourier analysis. In *Proceedings of the Twenty-Sixth Annual Symposium on Theory of Computing*, pages 253–262, 1994.

N. Bshouty and C. Tamon. On the Fourier spectrum of monotone functions. *Journal of the ACM*, 43(4):747–770, 1996.

N. Bshouty, J. Jackson, and C. Tamon. More efficient PAC learning of DNF with membership queries under the uniform distribution. In *Proceedings of the Twelfth Annual Conference on Computational Learning Theory*, pages 286–295, 1999.

N. Bshouty, E. Mossel, R. O'Donnell, and R. Servedio. Learning DNF from Random Walks. In *Proceedings of the 44th IEEE Symposium on Foundations on Computer Science*, pages 189–198, 2003.

D. Gamarnik. Extension of the PAC framework to finite and countable Markov chains. In *Proceedings of the 12th Annual Conference on Computational Learning Theory*, pages 308–317, 1999.

O. Goldreich. *Foundations of Cryptography: Volume 1, Basic Tools*. Cambridge University Press, New York, 2001.

O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the Association for Computing Machinery*, 33(4):792–807, 1986.

J. Håstad, R. Impagliazzo, L. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

J. Jackson. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. *Journal of Computer and System Sciences*, 55:414–440, 1997.

J. Jackson, A. Klivans, and R. Servedio. Learnability beyond AC^0. In *Proceedings of the 34th ACM Symposium on Theory of Computing*, pages 776–784, 2002.

M. Kharitonov. Cryptographic hardness of distribution-specific learning. In *Proceedings of the Twenty-Fifth Annual Symposium on Theory of Computing*, pages 372–381, 1993.

N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

S. Roch. On learning thresholds of parities and unions of rectangles in random walk models. *Random Structures and Algorithms*, 2006.

L. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

K. Verbeurgt. Learning DNF under the uniform distribution in quasi-polynomial time. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 314–326, 1990.

Journal of Machine Learning Research 8 (2007) 291-301 Submitted 1/06; Revised 7/06; Published 2/07

Comments on the "Core Vector Machines: Fast SVM Training on Very Large Data Sets"

Gaëlle Loosli GAELLE@LOOSLI.FR
Stéphane Canu STEPHANE.CANU@INSA-ROUEN.FR
LITIS, INSA de Rouen
Avenue de l'Université
76801 Saint-Etienne du Rouvray
France

Editor: Nello Cristianini

Abstract

In a recently published paper in JMLR, Tsang et al. (2005) present an algorithm for SVM called Core Vector Machines (CVM) and illustrate its performances through comparisons with other SVM solvers. After reading the CVM paper we were surprised by some of the reported results. In order to clarify the matter, we decided to reproduce some of the experiments. It turns out that to some extent, our results contradict those reported. Reasons of these different behaviors are given through the analysis of the stopping criterion.

Keywords: SVM, CVM, large scale, KKT gap, stopping condition, stopping criteria

1. Introduction

In a recently published paper in JMLR, Tsang et al. (2005) present an algorithm for SVM, called CVM. In this paper, some illustration of the CVM performances, compared to others solvers are shown. We have been interested in reproducing their results concerning the checkers problem because we knew that SimpleSVM (Vishwanathan et al., 2003) could handle such a problem more efficiently than reported in the paper. Tsang et al. (2005, Figure 3, page 379) show that CVM's training time is independent of the sample size and obtains similar accuracies to the other SVM solvers. We discuss those results through the analysis of the stopping criteria of the different solvers and the effects of hyper-parameters.

Indeed in SVMs, there are hyper-parameters: the slack trade-off (C) and the stopping tolerance (ε). For RBF kernels, the bandwidth (γ) is also an hyper-parameter and it can be estimated via the distance between points from opposite classes (Tsang et al., 2005, page 376). It can also be fixed using cross-validation. The slack trade-off is used to permit points to be misclassified. A small value for C allows many points to be misclassified by the solution while large value or infinite value forces the solution to classify well all the points (hard-margins). This hyper-parameter can be chosen with cross-validation or using knowledge on the the problem. Those two hyper-parameters are well-known and more details can be found in literature (Schölkopf and Smola, 2002). The stopping tolerance is most often left as a default value in the solvers. It corresponds to the precision required before stopping the algorithm and is strongly linked to the implementation. The value it takes is close to zero and we point out in this paper that it is not the value by itself that is important but the stopping criteria in which it is involved (Section 3).

Coming back to the comparison of solvers, our first experiment (Section 2) shows how different sets of hyper-parameters produce different behaviors for both CVM and SimpleSVM. We also give results with libSVM (Chang and Lin, 2001a) for the sake of comparison since SMO (Platt, 1999) is a standard in SVM resolution. Our results indicates that the choice of the hyper-parameters for CVM does not only influence the training time but also greatly the performance (more than for the other methods).

Section 3 aims at understanding what influences the most the CVM accuracy. We show that CVM uses a stopping criterion which may induce complex tuning and unexpected results. We also explore the behavior of CVM when C varies compared to libSVM and SimpleSVM.

In Section 4, our second experiment points out that the choice of the magnitude of the hyper-parameter C (which behaves as a regulator) is critical, more than the bandwidth for this problem. It is interesting to notice that each method has *modes* that are favorable and *modes* that are not. The last experiments enlightens those different modes.

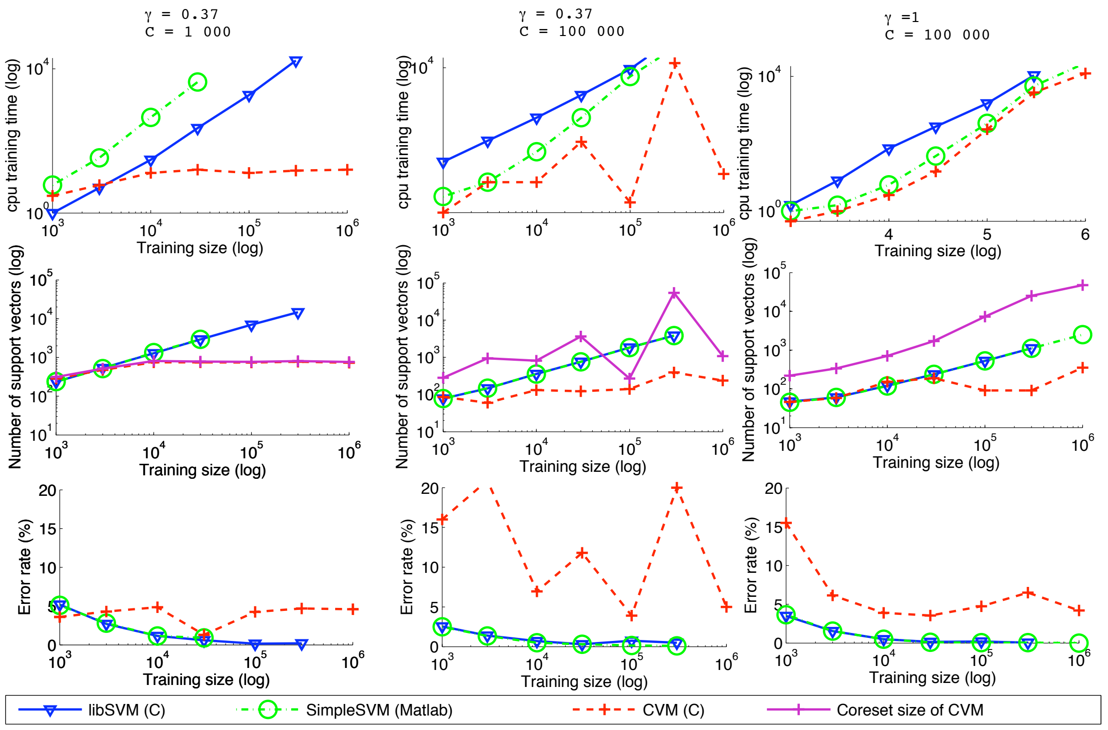


Figure 1: Comparison between CVM, libSVM and SimpleSVM on the checkers problem. For all figures, ∇ is for libSVM, o for SimpleSVM and $+$ for CVM. The three figures on the first column show the results while trying to reproduce Figure 3 (page 379) from the paper ($C = 1000$, $\gamma = 0.37$, *cf.* page 376, Section 6 and page 377, Section 6.1). The second column shows results with a larger C ($C = 100000$ and $\gamma = 0.37$) and the last shows results for $C = 100000$ and $\beta = 1$. Missing results are due to early termination because of not enough memory and/or the too long training time.

About the Reproducibility All experiments presented here can be reproduced with codes and data set that are all published.

- **CVM (Tsang et al., 2005)** Version 1.1 (in C) downloaded from `http://www.cs.ust.hk/~ivor/cvm.html` for Linux platform on the 3^{rd} of April 2006.
- **LibSVM (Chang and Lin, 2001a)** We used the release contained in CVM (here we should mention that a new version of libSVM exists (Fan et al., 2005), which would probably give some better results).
- **SimpleSVM (Vishwanathan et al., 2003; Loosli, 2004)** We used version 2.3 available at `http://asi.insa-rouen.fr/~gloosli/coreVSsimple.html`, written in Matlab.
- **Data** We have published the used data sets at `http://asi.insa-rouen.fr/~gloosli/coreVSsimple.html`, in both formats for libSVM and SimpleSVM.

Note that all results presented here are obtained using a stopping tolerance $\varepsilon = 10^{-6}$ as proposed in the CVM paper. Moreover, from now and for the rest of the paper, SimpleSVM will refer to the specific version implemented in the toolbox, as well as SMO will refer to the libSVM implementation.

2. Alternative Parameters For the Checkers

This section aims at verifying that SimpleSVM can treat one million points in problems like the checkers with adapted hyper-parameters. On the way, we also see that this leads to better performance than the one presented on Figure 3 page 379 of Tsang et al. (2005) and not only for SimpleSVM.

2.1 Original Settings

We used the parameters described in their paper ($\gamma = 0.37$ and $C = 1000$), γ being fixed to this value so the results can easily be reproduced (the setting of the paper relies on the average distance of couples of points on the training set). Note that $\gamma = 1/\beta$ corresponds to the hyper-parameters directly given to the toolboxes - $k(x_i, x_j) = \mathtt{exp}\big(-\gamma(x_i - x_j)^2\big)$ and that β is the notation used in the studied paper. Results are presented on Figure 1, first column.

2.2 Others Settings

Our motivation is that the checkers is a separable problem. First, this should reduce the number of support vectors and second, it should be possible to have a zero error in test. Thus we used in a second time some other hyper-parameters : the couple $\gamma = 0.37, C = 100000$ on the one hand and $\gamma = 1$ and $C = 100000$ on the second hand (found by cross-validation). Examples of results are presented on Figure 1, second and third columns.[1]

1. It turns out that these results are representative of the behaviors of the algorithms on this problem for most of randomly drawn training sets.

2.3 Results

- For the given settings, we can see similar behaviors as the one presented in the paper concerning training size, performance and speed. In particular, due to the large number of support vectors required for this setting (graph (b), Figure 1), SimpleSVM cannot be run over 30,000 data points,
- the possibility to run SimpleSVM or CVM until 1 million points depends on the settings,
- both SimpleSVM and libSVM achieve a 0 testing error (which means to us that the settings are more adapted) while CVM shows a really unstable behavior regarding testing error,
- for CVM, training time grows very fast for large values of C, due to huge intermediate number of active points (see Section 4), even if it still gives a sparse solution in the end,
- finally, the training error shows that CVM does not converge towards the solution for all hyper-parameters.

3. Study of CVM

Because Figure 1 shows very different behaviors of the algorithms, we focus in this part on the stopping criteria and on the role played by the different hyper-parameters. To do so we first compare CVM and SimpleSVM algorithms and then we point out that stopping criteria are different and may explain the results reported in Figure 1. We illustrate our claims on Figure 2 which shows that the magnitude of C is the key hyper-parameter to explore the different behaviors of the algorithms.

3.1 Algorithm

The algorithm presented in CVM uses MEB (Minimum Enclosing Balls) but can also be closely related to decomposition methods (such as SimpleSVM). Both methods consist in two alternating steps, one that defines groups of points and the other that computes the α (the Lagrangian multipliers involved in the dual of the SVM, see Schölkopf and Smola, 2002). In order to make clear that CVM and SimpleSVM are similar, we briefly explain those steps for each. Doing so we also enlighten their differences. Then we give the two algorithms and specify where stopping conditions apply.

3.1.1 DEFINING THE GROUPS

CVM: divides the database between *useless* and *useful* points. The useful points are designated as the *coreset* and correspond to all the points that are candidates to be support vectors. This set can only grow since no removing step is done. Among the points of the coreset, not all are support vectors in the end.

SimpleSVM: divides the database into three groups: I_w for the non bounded support vectors ($0 < \alpha_w < C$), I_C for the bounded points - misclassified or in the margins ($\alpha_C = C$) and I_0 which contains the non support vectors ($\alpha_0 = 0$).

3.1.2 FINDING THE α

CVM: solves the QP on the coreset only using SMO and thus obtains the α.

SimpleSVM: solves an optimization problem such that the optimality condition leads to a linear system. This linear system has α_w as unknown.

3.1.3 ALGORITHMS

CVM: uses the following steps:

Algorithme 1 CVM algorithm

```
1: initialize ▷ two points in the coreset, the first center of the ball, the first radius (see the original
   paper for details on the MEB)
2: if no point in the remaining points falls outside the $\varepsilon$-ball then
3:     stop (with $\varepsilon$-tolerance)                                    ▷ outer loop
4: end if
5: take the furthest point from the center of the current ball, add it to the core set
6: solve the QP problem on the points contained in the coreset (using SMO with warm start with
   $\varepsilon$-tolerance)                                                   ▷ inner loop
7: update the ball (center and radius)
8: go to second step
```

Two things require computational efforts there: the distance computation involved in steps 2 and 5, and solving the QP in step 6. For the distance computation, the authors propose a heuristics that consists of sampling 59 points instead of checking all the points, thus reducing greatly the time required. Doing so they ensure at 95% that the furthest point among those 59 points is part of the 5% of points the furthest from the center. For step 6, they use warm start feature of the SMO algorithm, doing a rank-one update of the QP.

SimpleSVM: uses the following steps:

Algorithme 2 SimpleSVM algorithm

```
1: initialize                                  ▷ two points in $I_w$, first values of $\alpha_w$
2: if no point in the remaining points is misclassified by the current solution then
3:     stop (with $\varepsilon$-tolerance)
4: end if
5: take the furthest misclassified point from the frontier, add it to $I_w$
6: update the linear system and retrieve $\alpha_w$
7: if all $0 < \alpha_w < C$ then
8:     go to second step
9: else
10:    remove violating point from $I_w$ (put it in $I_0$ or $I_c$ depending on the constraint it violates) and
       go to step 6
11: end if
```

Again, two things require computational efforts here: the classification involved in steps 2 and 5, and solving the linear system in step 6. For the classification, a heuristic is used that consists of taking the first violator found (yet all the points are eventually checked hence insuring an exact resolution). For step 6, a rank-one update is performed which is $O(n^2)$ instead of $O(n^3)$ for full resolution.

3.2 Stopping Criteria

We now consider the stopping criteria used in those algorithms. Indeed we have noted in the previous algorithms steps where a stopping criteria is involved (the ε-tolerance).

For the sake of clarity, we will also give details about the ν-SVM. Indeed, we will see that the CVM stopping criterion is analog to the ν-SVM one.

Notations and definitions Let $\mathbf{x}$ be our data labeled with $\mathbf{y}$. Let $k(.,.)$ be the kernel function. We will denote α the vector of Lagrangian multipliers involved in the dual problem of SVMs.

K is the kernel, such that $K_{ij} = y_i y_j k(x_i, x_j)$. As defined in Tsang et al. (2005, Eq. 17), $\tilde{K}$ is the modified kernel, such that $\tilde{K}_{ij} = y_i y_j k(x_i, x_j) + y_i y_j + \frac{\delta_{ij}}{C}$ ($\delta_{ij} = 1$ if $i = j$ and 0 otherwise). $\tilde{K}$ is used for the reformulation of the SVM as a MEB problem.

As defined in Tsang et al. (2005, page 370), $\tilde{\kappa} = \tilde{K}_{ii} = \kappa + 1 + \frac{1}{C}$ with $\kappa = K_{ii}$ (which is equal to 1 for the RBF kernel).

$\mathbf{1}$ stands for vectors of ones, $\mathbf{0}$ for vectors of zeros. All vectors are columns unless followed by a prime.

C-SVM type (SimpleSVM and SMO) The stopping criterion commonly used in classical SVM (C-SVM) is

$$\min(K\alpha_1 - \mathbf{1}) \geq -\varepsilon_1.$$

The two implementations SimpleSVM and libSVM (the C-SVM version) use similar stopping criteria, up to the biased term. This corresponds to the constraint of good classification. Here the magnitude of α, influenced by C, slightly modifies the *tightness* of the stopping condition. For SMO, a very strict stopping criteria causes a large number of iterations and slows down the training speed. This is the reason why it is usually admitted that SMO should not be run with large C values (and this idea is often extended to SVM independently of the solver, which is a wrong shortcut). On the other hand, SimpleSVM's training speed is linked to the number of support vectors (there is no direction search with a first order gradient to slow it down). For separable problems, large C produces sparser solutions, so SimpleSVM is faster with large C.

ν-SVM type In the ν-SVM setting (see Chen et al., 2005, for further details), the stopping criterion is

$$\min(K\alpha_2 - \rho_2\mathbf{1}) \geq -\varepsilon_2$$

where ρ_2 corresponds to the margin (see Chang and Lin, 2001b, for derivations). In order to compare this criterion to the previous one, it is possible to exhibit links between the two problems. Let say that for a given problem and a given ε_2, the ν-SVM optimal solution is α_2 and ρ_2. To obtain the same solution using the C-SVM, one can use the following relations: $C = 1/\rho_2$ (see Schölkopf and Smola, 2002, p209, prop. 7.6) and $\varepsilon_1 = \varepsilon_2/\rho_2$. The solution α_1 obtained by the C-SVM can also be compared to α_2 since we know that $0 \leq \alpha_2 \leq 1$ and $0 \leq \alpha_1 \leq C$. Thus there is $\alpha_2 = \alpha_1/C$. Thanks to those relations it is possible to express the C-SVM stopping criterion with the ν-SVM terms by setting

$$\varepsilon_1 = \varepsilon_2 C = \frac{\varepsilon_2}{\rho_2}.$$

With a small ε_2, a small margin found by the ν-SVM leads to a large C (for instance $1/\varepsilon_2$). The equivalent C-SVM should run with $\varepsilon_1 = 1$ which is very loose. Because of this scaling effect, comparison should be careful.

CVM CVM uses two overlapping loops. The outer one for the coreset selection and the inner one to solve the QP problem on the coreset. Thus CVM require two stopping criteria, one per loop.

Let us first consider the inner loop, that is, points that are constituting the coreset. The related multipliers α_3 are found using SMO. However, as mentioned in Tsang et al. (2005, Equation 5), the solved QP is similar to a ν-SVM problem. Hence the stopping condition is similar to the ν-SVM stopping condition. The major difference between a ν-SVM and the inner loop of CVM is the use of a modified kernel $\tilde{K}$ instead of K. The leading stopping criterion is the one of the outer loop.

The stopping criterion of the outer loop (equivalent to Eq. 25 of the paper) is:

$$\min\big((\tilde{K}\alpha_3)_{\mathcal{R}} - \rho_3\mathbf{1}\big) \geq -\varepsilon_3\tilde{\kappa}$$

where $\mathcal{R}$ indicates points that are outside the coreset. Note that if α_3 is the CVM solution for a given ε_3, since $\mathbf{1}'\alpha_3 = 1$, the equivalent solution α_1 given by a C-SVM is linked as follows : $\alpha_3 = \alpha_1/(\mathbf{1}'\alpha_1)$. Moreover, setting $\rho_3 = 1/(\mathbf{1}'\alpha_1)$, using those relations and expanding the kernel expression give:

$$\min\Big(\frac{\big((K+\mathbf{y}\mathbf{y}'+\frac{1}{C}Id)\alpha_1\big)_{\mathcal{R}} - \mathbf{1}}{\mathbf{1}'\alpha_1}\Big) \geq -\varepsilon_3(\kappa+1+\frac{1}{C}). \tag{1}$$

Here again thanks to those relations it is possible to express the C-SVM stopping criterion with the outer CVM terms by taking

$$\varepsilon_1 = \frac{\varepsilon_3(\kappa+1+\frac{1}{C}) + \big((\mathbf{y}\mathbf{y}'+\frac{1}{C}Id)\alpha_3\big)_\ell}{\rho_3}$$

where ℓ denotes the indice of the point minimizing the left-hand side part of (1). We can observe a similar scale effect to the one happening for ν-SVM.

Apart from the possibility to compare fairly to other solvers, let us study the behavior of CVM depending on C or on the training set size.

- The modified kernel hides a non-negligible regularizing effect. Indeed $1/C$ is added to the diagonal. Thus for small C, the kernel becomes well conditioned. Hence we can expect CVM to give better results with small C.
- From its ν-SVM like form, it also gets stricter condition for small C. This strict condition increases the number of steps and, as a side effect, the probability to check more points outside the coreset when sampling the 59 points.

Overall, for a small C, CVM should be slower than SMO (since it repeats SMO) and give accurate results.

- A large C, on the contrary, does not influence the kernel conditioning anymore.
- From the form of the stopping criterion, the condition is looser.
- The inner stopping condition is loose too. The looser it is, the fewer support vectors it takes.

Overall, we can expect less accurate results for large C.

As for the training set size, it plays on the scaling effect. The more points there are, the more important this effect is. As a consequence, the previous tendencies depending on C are more visible for large data sets. Finally, we would like to underline the fact that the stopping criterion for the inner loop is not clearly defined. This analysis is simplified by admitting that the inner stopping criterion is equivalent to the outer one.

4. Experimental Results

We now propose experiments illustrating the previous analysis. We first make the training size varies and then C. For all the experiments, we use $\varepsilon = 10^{-6}$.

Behavior of the algorithms when size is growing Figure 2 shows the results for experiments where the training size is growing for various sets of hyper-parameters. We observe:

- for a given C, results are similar in terms of training time, regardless of the kernel bandwidth,
- the best configuration for CVM is medium C regarding training time and small C regarding accuracy,
- for small C, CVM is the most accurate method (due to a regularization on the kernel),
- for large C, CVM accuracy tends to decrease.

It is clear from this experiment that we face different behaviors depending on the hyper-parameters. Each method has a favorable *mode* and those are not compatible. Since the bandwidth does not influence the trends of the obtained curves, we will fix it and focus now on variations of C.

Behavior of the algorithms when C is varying Figure 3 shows how the training changes depending on the value of C for each studied algorithm. The same experiment is conducted for two training sizes (10,000 points for the first column, 30,000 for the second), in order to point out that the training size amplifies sensitivity of CVM to its hyper-parameters. Each experiment reports the training time (top figure), the error rate (second figure), the number of support vectors and the size of the coreset (third figure) and the proportion of support vectors in the coreset for CVM (last figure), $C \in [10^{-2}, 10^{6}]$.

small C SimpleSVM fails because of not enough memory, libSVM is the fastest (loose stopping condition) and CVM the most accurate (regularized kernel). The coreset size corresponds almost exactly to the number of support vectors, which means that the selection criterion is accurate.

large C SimpleSVM is faster than libSVM (penalized by strict stopping condition) and both are as accurate. CVM gets large error rate and we see that the less the coreset set is related to the support vectors, the less accurate the method is. The point that is not so clear is the reason why the coreset grows that much and this is still to be explored.

5. Conclusion

In their conclusion, the authors claim that *"experimentally, it [CVM] is as accurate as existing SVM implementations"*. We show that is not always true and furthermore that it may not converge towards the solution.

Moreover we have shown that comparisons between CVM and usual SVM solvers should be careful since the stopping criteria are different. We have illustrated this point as well as the effect of magnitude of C on the training time. We have explained partly the behavior of CVM and shown that their results may be misleading, yet it still requires further exploration (particularly regarding the actual effects of the random sub-sampling).

As mentioned by Chang and Lin (2001b, conclusion), who are studying the ν-SVM (which has a similar stopping condition to the CVM, that is, very sensitive to the α magnitude), we would need

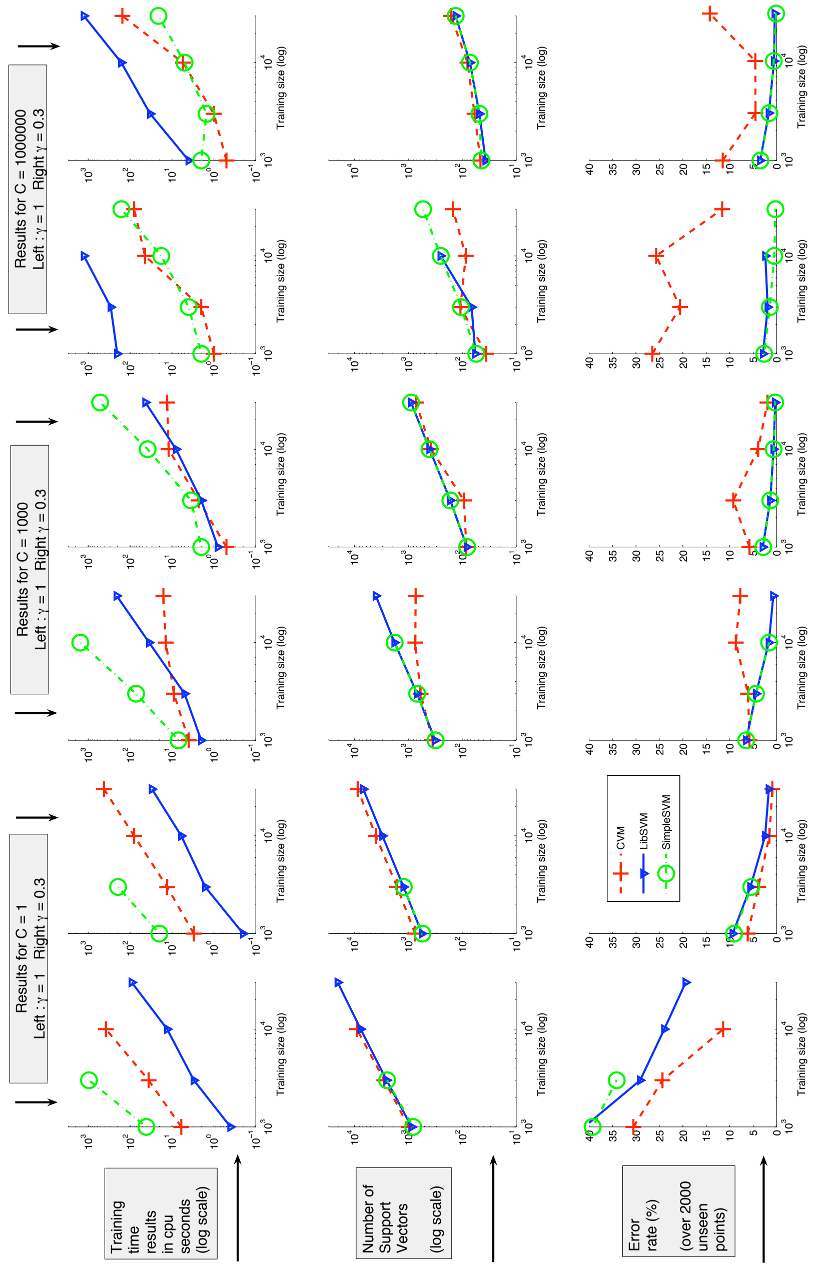


Figure 2: Experiments to compare results for different sets of hyper-parameters. Each column shows results for a couple of settings γ and C. For each column, the first row gives the training time for each algorithm. The second row shows the number of support vectors while the third gives the error rate. All reported results are obtained with a KKT tolerance $\varepsilon = 10^{-6}$. Note that those experiments are run up to 30,000 points only. Missing results are due to early termination because of not enough memory and/or the too long training time.

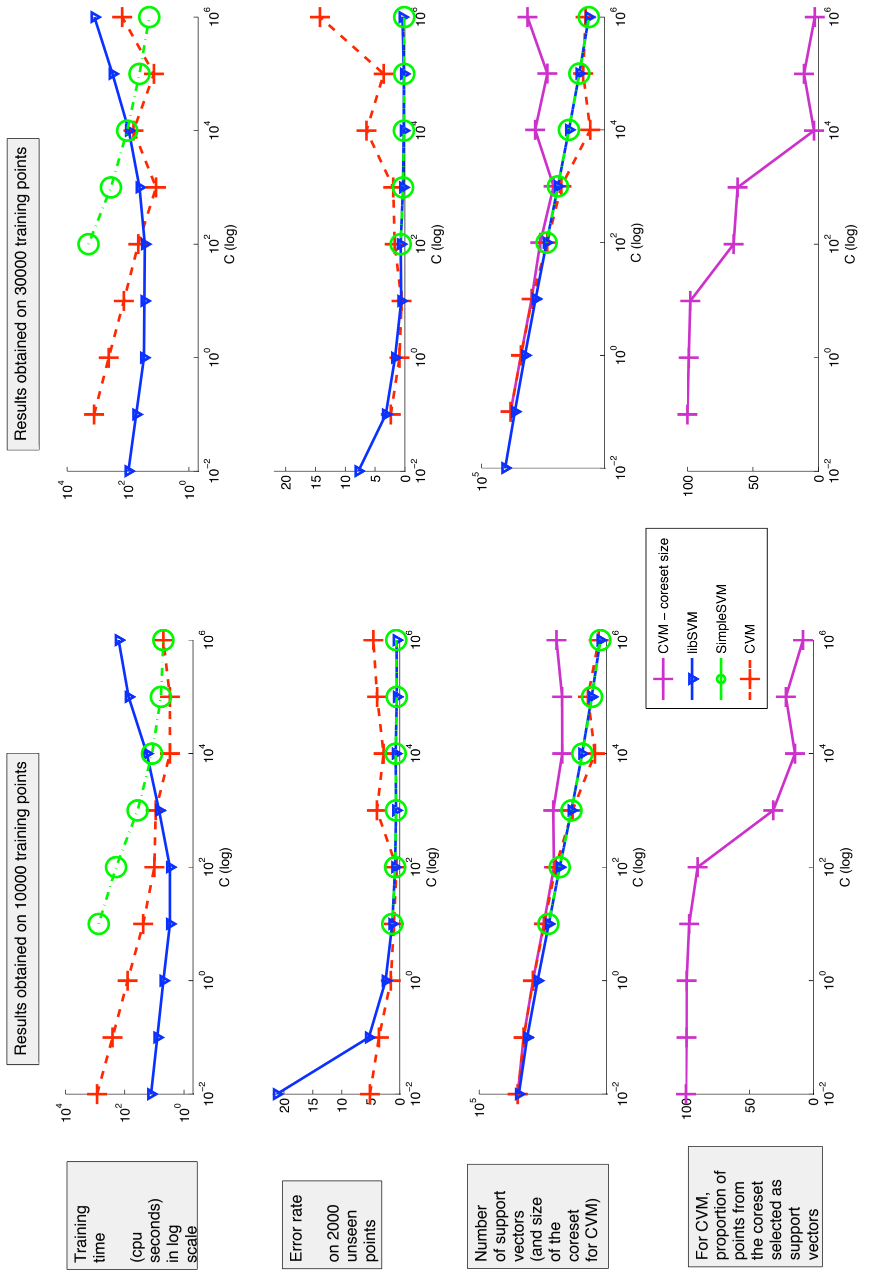


Figure 3: Experiments to illustrate the influence of hyper-parameter C for the different algorithms. First column reports experiments with 10,000 training points and the second uses 30,000 points. For each column, the top row gives the training time (log scale) depending on C while the second row shows the error rate achieved on 2000 unseen points. The third row gives the number of support vector as well as the size of the coreset for CVM. The bottom row reports the proportion of support vectors selected in the coreset for CVM. All experiments are done with $\gamma = 1$ and $\varepsilon = 10^-6$.

here an adaptive ε in order to avoid unstable behaviors but it can not be done easily. Furthermore, one who uses SVM should be aware that each method has favorable and non favorable ranges of hyper-parameters.

Finally, we acknowledge that a fast heuristic which achieves loose yet still acceptable accuracy is interesting and useful for very large databases (by itself or as a starting point for more accurate methods) but to use it with confidence, one has to know when the method is indeed accurate enough. Hence CVM may benefit from being presented along with its limits.

Acknowledgments

We would like to thank very much the anonymous reviewers for helpful comments on the stopping conditions. This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

References

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines, 2001a. URL `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Chih-Chung Chang and Chih-Jen Lin. Training ν-support vector classifiers: Theory and algorithms. *Neural Computation*, (9):2119–2147, 2001b.

Pai-Hsuen Chen, Chih-Jen Lin, and Bernhard Schölkopf. A tutorial on ν-support vector machines. *Applied Stochastic Models in Business and Industry*, 21(2):111–136, 2005.

Rong-En Fan, Pai-Hsuen Chen, and Chih-Jen Lin. Working set selection using second order information for training support vector machines. *Journal of Machine Learning Research*, 2005.

Gaëlle Loosli. Fast svm toolbox in Matlab based on SimpleSVM algorithm, 2004. `http://asi.insa-rouen.fr/~gloosli/simpleSVM.html`.

John Platt. Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advanced in Kernel Methods - Support Vector Learning*, pages 185–208. MIT Press, 1999.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. MIT Press, 2002.

Ivor W. Tsang, James T. Kwok, and Pak-Ming Cheung. Core vector machines: Fast SVM training on very large data sets. *J. Mach. Learn. Res.*, 6:363–392, 2005.

S. V. N Vishwanathan, Alexander J. Smola, and M. Narasimha Murty. SimpleSVM. *Proceedings of the Twentieth International Conference on Machine Learning*, 2003.

Journal of Machine Learning Research 8 (2007) 303-321 Submitted 2/06; Revised 10/06; Published 2/07

General Polynomial Time Decomposition Algorithms

Nikolas List NLIST@LMI.RUB.DE
Hans Ulrich Simon SIMON@LMI.RUB.DE
Fakultät für Mathematik
Ruhr-Universität Bochum
44780 Bochum
Germany

Editor: Thorsten Joachims

Abstract

We present a general decomposition algorithm that is uniformly applicable to every (suitably normalized) instance of Convex Quadratic Optimization and efficiently approaches an optimal solution. The number of iterations required to be within ε of optimality grows linearly with $1/\varepsilon$ and quadratically with the number m of variables. The working set selection can be performed in polynomial time. If we restrict our considerations to instances of Convex Quadratic Optimization with at most k_0 equality constraints for some fixed constant k_0 plus some so-called box-constraints (conditions that hold for most variants of SVM-optimization), the working set is found in linear time. Our analysis builds on a generalization of the concept of rate certifying pairs that was introduced by Hush and Scovel. In order to extend their results to arbitrary instances of Convex Quadratic Optimization, we introduce the general notion of a rate certifying q-set. We improve on the results by Hush and Scovel (2003) in several ways. First our result holds for Convex Quadratic Optimization whereas the results by Hush and Scovel are specialized to SVM-optimization. Second, we achieve a higher rate of convergence even for the special case of SVM-optimization (despite the generality of our approach). Third, our analysis is technically simpler.

We prove furthermore that the strategy for working set selection which is based on rate certifying sets coincides with a strategy which is based on a so-called "sparse witness of sub-optimality". Viewed from this perspective, our main result improves on convergence results by List and Simon (2004) and Simon (2004) by providing convergence rates (and by holding under more general conditions).

Keywords: convex quadratic optimization, decomposition algorithms, support vector machines

1. Introduction

Support vector machines (SVMs) introduced by Vapnik and co-workers (Boser et al., 1992; Vapnik, 1998) are a promising technique for classification, function approximation, and other key problems in statistical learning theory. In this paper, we consider the optimization problems that are induced by SVMs. These SVM-optimization problems (SVO) are special cases of Convex Quadratic Optimization (CQO).

The difficulty of solving problems of this kind is the density of the matrix that represents the "quadratic part" of the cost function. Thus, a prohibitive amount of memory is required to store the matrix and traditional optimization algorithms (such as Newton, for example) cannot be directly applied. Several authors have proposed (different variants of) a decomposition method to overcome this difficulty (Osuna et al., 1997; Joachims, 1998; Platt, 1998; Saunders et al., 1998; Mangasarian and Musicant, 1999, 2000, 2001; Chang et al., 2000; Keerthi et al., 2000, 2001; Keerthi and Gilbert, 2002; Lin, 2001b, 2002a,b; Laskov, 2002; Chang and Lin, 2001; Hsu and Lin, 2002; Liao et al., 2002; Hush and Scovel, 2003; List and Simon, 2004; List, 2004; Chen et al., 2005). Given an instance of CQO, this method keeps track of a current feasible solution which is iteratively improved. In each iteration the variable indices are split into a "working set" $I \subseteq \{1,\dots,m\}$ and its complement $J = \{1,\dots,m\} \setminus I$. Then, the simplified instance with the variables $x_i, i \in I$, is solved, thereby leaving the values for the remaining variables x_j, $j \in J$, unchanged. The success of the method depends in a quite sensitive manner on the policy for the selection of the working set I (whose size is typically much smaller than m). Ideally, the selection procedure should be computationally efficient and, at the same time, effective in the sense that the resulting feasible solutions become cost-optimal in the limit (with high speed of convergence).

Our results and their relation to previous work: Hush and Scovel (2003) were concerned with SVO. They introduced the notion of an "α-rate certifying pair" and showed that every decomposition algorithm for SVO that always inserts an α-rate certifying pair in its current working set comes within ε of optimality after $O(1/(\varepsilon\alpha^2))$ iterations. Building on a result by Chang et al. (2000), they presented furthermore an algorithm that constructs an $1/m^2$-rate certifying pair in $O(m \log m)$ steps.[1] Combining these results, we see that the decomposition algorithm of Hush and Scovel for problem SVO is within ε of optimality after $O(m^4/\varepsilon)$ iterations.

In this paper we present an extension of (and an improvement on) this result. We first define the general notion of an α-rate certifying q-set and show (with a simplified analysis) that it basically fits the same purpose for CQO as the α-rate certifying pair for SVO, where the number of iterations needed to be within ε of optimality is proportional to $q/(\varepsilon\alpha^2)$. We present a general decomposition algorithm that is uniformly applicable to every (suitably normalized) instance of CQO. Given an instance with k equality constraints and m variables, it finds an $1/m$-rate certifying $(k+1)$-set in polynomial time.[2] Combining these results, we are within ε of optimality after $O(km^2/\varepsilon)$ iterations of our decomposition algorithm. The problem SVO (considered by Hush and Scovel) has only one equality constraint. Plugging in $k = 1$ in our general result, we arrive at an upper bound on the number of iterations that improves on the bound obtained by Hush and Scovel by factor m^2. The analysis of Hush and Scovel (2003) builds on an earlier analysis of conditional gradient algorithms given by Dunn (1979). For this part of the analysis, we will present simpler arguments.

List and Simon (2004) suggested a strategy for working set selection that is based on a so-called "sparse witnesses of sub-optimality". They were able to prove that this strategy leads to feasible solutions of minimum cost in the limit. To this end, they had to impose a technical restriction on the cost function. Their result states the mere convergence without providing any convergence rates. In this paper, we prove (by means of Linear Programming duality) that the strategy for working set selection which is based on rate certifying sets coincides with the strategy of List and Simon (2004). Viewed from this perspective, our main result improves on the convergence result by List and Simon (2004) (and on a related result by Simon (2004)) by providing convergence rates (and by holding under more general conditions).

There are some alternatives to the approach we follow here. A paper by Lin (2001a) seems to imply the following quite strong result for SVO (although not stating it explicitly): decomposition algo-

1. Time bound $O(m \log m)$ can be improved to $O(m)$ by using the method of Simon (2004).
2. Moreover, the algorithm can be implemented such as to find this set even in linear time when we restrict its application to instances of CQO with at most k_0 equality constraints for some fixed constant k_0. If we restrict its application to SVO, we may use the highly efficient method of Simon (2004).

rithms following the approach of maximally KKT-violating pairs are within ε of optimality after only $O(\log 1/\varepsilon)$ iterations.[3] However, the analysis is specialized to SVO. Furthermore it has to assume strict convexity of the objective function and some non-degeneracy conditions. The convergence rate is only given in terms of ε whereas the dependence on problem parameters (like, for example, m) is not clarified.
Another algorithm (related to but different from decomposition algorithms) is SimpleSVM (Vishwanthan et al., 2003) which tries to iteratively include the support vectors in the working set. Assuming strict convexity of the objective function, Vishwanthan et al. (2003) claim a linear convergence of the method (but do neither give a complete proof nor exhibit the dependence on the various parameters explicitly). The main difference between SimpleSVM and decomposition algorithms is the size of the working set which can grow-up to the number of support vectors in the former case and is kept constant in the latter. Note that the number of support vectors is particularly large on noisy data.

2. Preliminaries

After fixing some notation (Section 2.1), we briefly recall the main result by Hush and Scovel (2003) about SVO and rate certifying pairs (Section 2.2) and the main result by List and Simon (2004) about CQO and sparse witnesses of sub-optimality (Section 2.3).

2.1 Notations and Basic Facts

We are mainly concerned with the problem "Convex Quadratic Optimization with box-constraints". It is denoted simply as CQO in this paper and is formally given as follows:

Definition 1 (CQO) *An instance $\mathcal{P}$ of CQO is given by*

$$\min_x f(x) = \frac{1}{2}x^\top Qx + w^\top x \text{ s.t. } Ax = b, l \le x \le r \quad ,$$

where

- *$Q \in \mathbb{R}^{m\times m}$ is a symmetric positive semi-definite matrix over the reals and $w \in \mathbb{R}^m$. That means $f(x)$ is a convex quadratic cost function in m scalar variables,*
- *$A \in \mathbb{R}^{k\times m}$ and $b \in \mathbb{R}^k$ such that $Ax = b$ represents k linear equality constraints,*
- *$l, r \in \mathbb{R}^m$ and $l \le x \le r$ is the short-notation for the "box-constraints"*

$$\forall i = 1, \ldots, m : l_i \le x_i \le r_i \quad .$$

In this paper, we will sometimes express $f(x')$ by means of the Taylor-expansion around x:

$$f(x') = f(x) + \nabla f(x)^\top (x' - x) + \frac{1}{2}(x' - x)^\top Q(x' - x) \quad ,$$

3. See the work of Chen et al. (2005, 2006) for a generalization of this result to similar but more general policies for working set selection.

where $\nabla f(x) = Qx + w$. This can be rewritten as follows:

$$f(x) - f(x') = \nabla f(x)^\top (x - x') - \frac{1}{2}(x - x')^\top Q(x - x') \ . \tag{1}$$

Note that $(x - x')^\top Q(x - x') \geq 0$ because Q is positive semi-definite.

In the sequel,

$$R(\mathcal{P}) = \{x \in \mathbb{R}^m | \ Ax = b, l \leq x \leq r\}$$

denotes the compact and convex set of feasible points for $\mathcal{P}$. The well-known first-order optimality condition for convex optimization (see, for example, Boyd and Vandenberghe, 2004) (valid for an arbitrary convex cost function) states that $x \in R(\mathcal{P})$ is an optimal feasible solution iff

$$\forall x' \in R(\mathcal{P}) : \nabla f(x)^\top (x' - x) \geq 0 \ .$$

We briefly note that any instance of the *general* convex quadratic optimization problem with cost function $f(x)$, linear equality constraints, linear inequality constraints (not necessarily in the form of box-constraints) and a compact region of feasible points can be transformed into an equivalent instance of CQO because we may convert the linear inequalities into linear equations by introducing non-negative slack variables. By the compactness of the region of feasible points, we may also put a suitable upper bound on each slack variable such that finally all linear inequalities take the form of box-constraints.

We now define a subproblem of CQO, denoted as SVO in this paper, that is actually one of the most well studied SVM-optimization problems:

Definition 2 (SVO) *An instance $\mathcal{P}_0$ of SVO is given by*

$$\min_x f(x) \text{ s.t. } y^\top x = 0 \ , \ l \leq x \leq r \ ,$$

where $f(x), l, r$ are understood as in Definition 1 and $y \in \{-1, 1\}^m$ is a vector whose components represent binary classification labels.

The main difference between SVO and the general problem CQO is that SVO has only a single equality constraint. Furthermore, this equality constraint is of a special form.

We are now prepared to introduce (informally) the notion of "decomposition algorithms". A *decomposition algorithm for CQO* with working sets of size at most q (where we allow that q depends on k) proceeds iteratively as follows: given an instance $\mathcal{P}$ of CQO and a feasible solution $x \in R(\mathcal{P})$ (chosen arbitrarily in the beginning), a so-called working set $I \subseteq \{1, \ldots, m\}$ of size at most q is selected. Then x is updated by an optimal solution for the simplified instance with variables x_i, $i \in I$ (leaving the values x_j with $j \notin I$ unchanged). *Decomposition algorithms for SVO* are defined analogously. The policy for working set selection is a critical issue that we discuss in the next sections.

Notational Conventions:

- The parameters m, k, f, A, y, b, l, r are consistently used in this paper as the components of an instance of CQO or SVO. Similarly, parameter q (possibly dependent on k) always represents the (maximal) size of the working set.

- L_{max} and S_{max} are two more parameters that we associate with an instance of CQO or SVO (where L_{max} depends also on q). L_{max} denotes the largest among the eigenvalues of all the principal $(q \times q)$-sub-matrices of Q. $S_{max} := \max_{1 \le i \le m}(r_i - l_i)$ denotes the maximum side length of the box spanned by l and r.

- For a decomposition algorithm $\mathcal{A}$, we denote the current feasible solution obtained after n iterations by x^n (such that x^0 is the feasible solution $\mathcal{A}$ starts with[4]). x^* always denotes an optimal feasible solution such that

$$\Delta_n := f(x^n) - f(x^*) \tag{2}$$

denotes the difference between the value of the current solution and the optimal value.

2.2 Rate Certifying Pairs

Let us consider problem SVO from Definition 2 along with a problem instance $\mathcal{P}_0$. Let $x \in R(\mathcal{P}_0)$ be a feasible solution and $x^* \in R(\mathcal{P}_0)$ an optimal feasible solution. In the sequel, we will often use the following first-order approximation of the maximal distance (with regard to the value of the objective function) between a given point x and any other feasible solution x'

$$\sigma(x) \quad := \quad \sup_{x' \in R(\mathcal{P}_0)} \nabla f(x)^\top (x - x') \ .$$

As already noted by Hush and Scovel (2003), the following holds:[5]

$$f(x) - f(x^*) \le \nabla f(x)^\top (x - x^*) \le \sigma(x) \ . \tag{3}$$

In other words, $f(x)$ is always within $\sigma(x)$ of optimality. Note that $\sigma(x^*) = 0$ (which immediately follows from the first-order optimality condition). Thus, $\sigma(x) > 0$ if and only if x is sub-optimal.

Since we are dealing with working sets whose size is bounded by a (small) parameter q, it is natural to restrict the range of x' to feasible solutions that differ from x in at most q coordinates. For $q = 2$, this leads to the following definition:

$$\sigma(x|i_1,i_2) \quad := \quad \sup_{x' \in R(\mathcal{P}_0):x'_i = x_i \text{ for } i \ne i_1,i_2} \nabla f(x)^\top (x - x') \ .$$

The following notion is crucial: (i_1,i_2) is called an *α-rate certifying pair* for x if

$$\sigma(x|i_1,i_2) \ge \alpha(f(x) - f(x^*)) \ .$$

Let $\vec{\alpha}$ be a function in m with strictly positive values. An *$\vec{\alpha}$-rate certifying algorithm* is a decomposition algorithm for SVO that, for every m and every input instance $\mathcal{P}_0$ with m variables, always includes an $\vec{\alpha}(m)$-rate certifying pair in the current working set. As mentioned already in the introduction, the main results by Hush and Scovel (2003) are as follows:

4. The problem of finding an initial feasible solution can be cast as an instance of Linear Programming. For an instance of SVO, an initial guess usually is the zero vector.
5. The first inequality follows from (1) and the positive semi-definiteness of Q; the second-one is trivial.

Theorem 3 (Hush and Scovel (2003)) *1. Let $\mathcal{A}$ be an $\vec{\alpha}$-rate certifying algorithm. Consider any instance $\mathcal{P}_0$ of SVO with, say, m variables. Let L_{max} and S_{max} be the quantities associated with $\mathcal{P}_0$.*[6] *For sake of brevity, let $\alpha = \vec{\alpha}(m)$. Then, $\mathcal{A}$ is within ε of optimality after*

$$1 + \frac{2\max\{1, 2S_{max}^2\}}{\alpha}\left(\frac{\max\{1, \alpha\Delta_0/L_{max}\}L_{max}}{\alpha\varepsilon} - 1\right) = O\left(\frac{L_{max}(1+S_{max})^2}{\alpha^2\varepsilon} + \frac{\Delta_0(1+S_{max})^2}{\alpha\varepsilon}\right)$$

iterations.

2. *For function $\vec{\alpha}$ given by $\vec{\alpha}(m) = 1/m^2$, there exists an $\vec{\alpha}$-rate certifying algorithm. It constructs a working set (given $\mathcal{P}_0$ and a sub-optimal feasible solution x) in $O(m\log m)$ steps (or in $O(m)$ steps when the method of Simon (2004) is applied). Furthermore, it is within ε of optimality after*

$$O\left(\frac{L_{max}(1+S_{max})^2 m^4}{\varepsilon} + \frac{\Delta_0(1+S_{max})^2 m^2}{\varepsilon}\right)$$

iterations.

2.3 Sparse Witnesses of Sub-optimality

Consider the situation where we apply the decomposition method to a problem instance $\mathcal{P}$ of the general problem CQO from Definition 1. As suggested by List and Simon (2004), the selection of a working set can be guided by a function family $(\mathcal{C}_I(x))_{I\subseteq\{1,\ldots,m\}}$ as follows:

Strict Rule: Given the current feasible solution x, choose the next working set $I \subseteq \{1,\ldots,m\}$ of size at most q such as to maximize $\mathcal{C}_I(x)$.

Relaxed Rule: Given the current feasible solution x, choose the next working set $I \subseteq \{1,\ldots,m\}$ of size at most q such as to maximize $\mathcal{C}_I(x)$ up to a factor of $0 < \rho < 1$.

List and Simon (2004) say that $(\mathcal{C}_I(x))$ is a *q-sparse witness of sub-optimality* if it satisfies the following conditions:

(C1) For each $I \subseteq \{1,\ldots,m\}$ such that $|I| \le q$, $\mathcal{C}_I(x)$ is continuous on $R(\mathcal{P})$.

(C2) If $|I| \le q$ and x' is an optimal solution for the subproblem induced by the current feasible solution x and working set I, then $\mathcal{C}_I(x') = 0$.

(C3) If x is not an optimal solution for $\mathcal{P}$, then there exists an $I \subseteq \{1,\ldots,m\}$ such that $|I| \le q$ and $\mathcal{C}_I(x) > 0$.

The following is shown by List and Simon (2004) and Simon (2004):

1. The function family

$$C_I(x) := \inf_{\lambda\in\mathbb{R}^k}\left(\sum_{i\in I}(x_i - l_i)\max\{0, \nabla f(x)_i - A_i^\top\lambda\} + (r_i - x_i)\max\{0, A_i^\top\lambda - \nabla f(x)_i\}\right) \quad (4)$$

6. See our notational conventions in Section 2 for a definition (where $q = 2$).

is a q-sparse witness of sub-optimality provided that $q \geq k+1$. Here, A_i denotes the i'th column of A and $A_i^\top$ its transpose.

2. The working set selection problem induced by the function family $C_I(x)$ from (4) is NP-complete if we insist on the strict rule. But it is solvable in polynomial time if we follow the relaxed rule with a constant $0 < \rho < 1$.

3. Let $q \geq k+1$. Assume that $Q \in \mathbb{R}^{m \times m}$ has positive definite sub-matrices $Q_{I,I}$ for all subsets $I \subseteq \{1, \dots, m\}$ of size at most q. If $(C_I(x))$ is a q-sparse witness of sub-optimality and the working set is always selected according to the relaxed rule, then the resulting feasible solutions become cost-optimal in the limit.

Note that the latter convergence result holds especially for the family $(C_I(x))$ from (4). However, it merely states convergence without guaranteeing any convergence rate. Furthermore, it requires the above assumption of positive definite sub-matrices $Q_{I,I}$. The results in the next section improve on this as follows:

- They do provide a convergence rate.
- They do not require the technical assumption about positive definite sub-matrices of Q.

3. Rate Certifying Sets and the Main Theorem

We now turn our attention to the main results of this paper. In Section 3.1, we present the new notion of rate certifying sets and state the main convergence theorem whose proof is given in Sections 3.2 and 3.3. In Section 3.4, we reveal the relation between rate certifying sets and sparse witnesses of sub-optimality.

3.1 Rate Certifying Sets

The notion of rate certifying sets generalizes the notion of rate certifying pairs from Section 2.2. The definition of $\sigma(x)$ is easily extended to any instance $\mathcal{P}$ of CQO:

$$\sigma(x) := \sup_{x' \in R(\mathcal{P})} \nabla f(x)^\top (x - x') \ . \tag{5}$$

Clearly, inequality (3) and the subsequent comments are still valid without any change. However, since CQO deals with several equality constraints, one can in general not expect to find rate certifying *pairs*. Instead, the following general definition for $I \subseteq \{1, \dots, m\}$ will prove useful:

$$\sigma(x|I) := \sup_{x' \in R(\mathcal{P}): x'_i = x_i \text{ for } i \notin I} \nabla f(x)^\top (x - x') \ . \tag{6}$$

I is called an *α-rate certifying q-set* if $|I| \leq q$ and

$$\sigma(x|I) \geq \alpha(f(x) - f(x^*)) \ . \tag{7}$$

I is called a *strong α-rate certifying q-set* (implying that it is an α-rate certifying q-set) if $|I| \leq q$ and

$$\sigma(x|I) \geq \alpha\sigma(x) \ .$$

Let $\vec{\alpha}$ be a function in m with strictly positive values and let $\vec{q}$ be a function in k whose values are strictly positive integers. A (strong) $(\vec{\alpha},\vec{q})$-*rate certifying algorithm* is a decomposition algorithm for CQO that, for every m,k and any problem instance $\mathcal{P}$ with m variables and k equality constraints, includes a (strong) $\vec{\alpha}(m)$-rate certifying $\vec{q}(k)$-set in the current working set. With these notations, the following holds:

Theorem 4 *1. Let $\mathcal{A}$ be an $(\vec{\alpha},\vec{q})$-rate certifying algorithm. Consider an instance $\mathcal{P}$ of CQO with, say, m variables and k equality constraints. For sake of brevity, let $\alpha = \vec{\alpha}(m)$, $q = \vec{q}(k)$, and let L_{max}, S_{max} be the quantities associated with $\mathcal{P}$ and q. Then, $\mathcal{A}$ is within ε of optimality after*

$$\left\lceil \frac{2qL_{max}S_{max}^2}{\alpha^2\varepsilon} \right\rceil + \max\left\{0, \left\lceil \frac{2}{\alpha}\ln\left(\frac{\Delta_0}{\varepsilon}\right) \right\rceil \right\} \tag{8}$$

iterations. Moreover, if $qL_{max}S_{max}^2 \le \varepsilon\alpha$, then $\max\{0, \lceil 2\ln(\Delta_0/\varepsilon)/\alpha \rceil\}$ *iterations (the second term in (8)) are enough.*

2. For functions $\vec{\alpha},\vec{q}$ given by $\vec{\alpha}(m) = 1/m$ and $\vec{q}(k) = k+1$, there exists a strong $(\vec{\alpha},\vec{q})$-rate certifying algorithm $\mathcal{A}$.[7] It constructs a working set (given $\mathcal{P}$ and a sub-optimal feasible solution x) in polynomial time. Moreover, if we restrict its application to instances of CQO with at most k_0 equality constraints for some fixed constant k_0, there is a linear time bound for the construction of the working set.[8]

3. The algorithm $\mathcal{A}$ from the preceding statement is within ε of optimality after

$$\left\lceil \frac{2(k+1)m^2L_{max}S_{max}^2}{\varepsilon} \right\rceil + \max\left\{0, \left\lceil 2m\ln\left(\frac{\Delta_0}{\varepsilon}\right) \right\rceil \right\}$$

iterations. Moreover, if $(k+1)L_{max}S_{max}^2 \le \varepsilon/m$, then $\max\{0, \lceil 2m\ln(\Delta_0/\varepsilon) \rceil\}$ *iterations are enough.*

Clearly, the third statement in Theorem 4 follows directly from the first two statements (which will be proven in Subsections 3.2 and 3.3, respectively).

A few comments on Theorem 4 are in place here. One might be tempted to think that an $(\vec{\alpha},\vec{q})$-rate certifying algorithm decreases (an upper bound on) the distance between the current feasible solution and the best feasible solution (with regard to the objective value) roughly by factor $1-\alpha$ (for $\alpha := \vec{\alpha}(m)$). If such a "contraction" took place, we would be within ε of optimality after only $O(\log(1/\varepsilon)/\alpha)$ iterations. This is however spurious thinking because the σ-function is *not* concerned with this distance itself but rather with a first-order approximation of it. We will see in the proof of Theorem 4 that a run of an $(\vec{\alpha},\vec{q})$-rate certifying algorithm can be decomposed into two phases. As long as the distance from the optimal value is large in comparison to (an upper bound on) the second order terms (phase 1), a contraction by factor $1-\alpha/2$ takes place. However when we come closer to the optimal value (phase 2), the effect of the neglected second order terms becomes more significant and the convergence slows down (at least within our perhaps somewhat pessimistic analysis). Phase 1 leads to the term $\max\{0, \lceil 2\ln(\Delta_0/\varepsilon)/\alpha \rceil\}$ in (8) whereas phase 2 leads to the term $\lceil (2qL_{max}S_{max}^2)/(\alpha^2\varepsilon) \rceil$.

7. As for the subproblem SVO and the special case $q = 2$, $\vec{\alpha}(m) = 1/m$ can be slightly improved to $\vec{\alpha}(m) = 1/(m-1)$ and this is optimal (Hush et al., 2006).

8. If we restrict its application to instances of SVO, we may use the highly efficient method of Simon (2004).

Note, that our analysis relies on the assumption, that the subproblem is solved exactly. As we always deal with rational numbers in practice, the solution computed will only achieve the optimum up to a certain precision ε. But as we can choose this ε arbitrarily small we will ignore this complication.

3.2 Proof of the 1st Statement in Theorem 4

We will use the notation introduced in Theorem 4 and consider an arbitrary iteration of the $(\vec{\alpha},\vec{q})$-rate certifying algorithm $\mathcal{A}$ when it is applied on input $\mathcal{P}$. To this end, let x denote a feasible but sub-optimal solution, and let x^* be an optimal feasible solution. Let I be the subset of the working set that satisfies $|I| \leq q$ and (7). Let x' be a feasible solution that satisfies $x_i = x'_i$ for every $i \notin I$ and

$$\sigma(x|I) = \nabla f(x)^\top (x - x') = \nabla f(x)^\top d \ \ , \tag{9}$$

where $d = x - x'$. Combining (7) with (9), we get

$$\nabla f(x)^\top d \geq \alpha(f(x) - f(x^*)) = \alpha\Delta \ \ , \tag{10}$$

where $\Delta = f(x) - f(x^*) \geq 0$. For some parameter $0 \leq \lambda \leq 1$ (suitably chosen later), consider the feasible solution

$$x'' = x - \lambda d$$

on the line segment between x and x'. Taylor-expansion around x allows to relate $f(x)$ and $f(x'')$ as follows:

$$f(x) - f(x'') = \lambda \nabla f(x)^\top d - \lambda^2 \frac{1}{2} d^\top Q d \ \ .$$

Note that x'' (like x') satisfies $x''_i = x_i$ for every $i \notin I$. Thus, $x'' = x - \lambda d$ is a feasible solution that coincides with x outside the current working set. Thus, $\mathcal{A}$ (finding an optimal feasible solution that coincides with x outside the working set) achieves in the next iteration a "cost reduction" of at least $f(x) - f(x'')$.[9] x'' depends on the parameter $0 \leq \lambda \leq 1$. In the sequel, we tune parameter λ such as to obtain a "large" cost reduction. To be on the safe side, we will however perform a "pessimistic analysis" where we substitute worst case bounds for $\nabla f(x)^\top d$ and $d^\top Q d$ respectively. Clearly

$$\max_{0 \leq \lambda \leq 1} \left(\lambda \nabla f(x)^\top d - \lambda^2 \frac{1}{2} d^\top Q d \right) \geq \max_{0 \leq \lambda \leq 1} \left(\lambda B - \frac{1}{2} \lambda^2 B' \right)$$

for any lower bound B on $\nabla f(x)^\top d$ and any upper bound B' on $d^\top Q d$. According to (10), $\alpha\Delta$ can serve as a lower bound on $\nabla f(x)^\top d$. It is easily seen that the following parameter U can serve as an upper bound on $d^\top Q d$:

$$U := q L_{max} S_{max}^2 \geq d^\top Q d \ \ . \tag{11}$$

This immediately follows from the definition of L_{max} and S_{max} and from the fact that d has at most q non-zero components. We conclude from this discussion that, for every $0 \leq \lambda \leq 1$, $\mathcal{A}$ achieves in the next iteration a cost reduction of at least

$$h(\lambda) := \lambda \alpha \Delta - \frac{1}{2} \lambda^2 U \ \ .$$

9. "Achieving a cost reduction of a in the next iteration" means that the next iteration decreases the distance between the value of the current feasible solution and the value of an optimal feasible solution by at least a.

It is easily seen that function $h(\lambda)$ is maximized by setting

$$\lambda := \begin{cases} 1 & \text{if } \Delta > U/\alpha \\ \alpha\Delta/U & \text{if } \Delta \leq U/\alpha \end{cases} \quad .$$

Case 1 $\Delta > U/\alpha$. Then $\mathcal{A}$ achieves a cost reduction of at least

$$h(1) = \alpha\Delta - \frac{1}{2}U > \alpha\Delta/2 \ \ .$$

Thus the difference $\Delta = f(x) - f(x^*)$ will shrink after the next iteration to

$$\Delta - \frac{\alpha}{2}\Delta = \Delta\left(1 - \frac{\alpha}{2}\right) \ \ ,$$

or to a smaller value, which is a proper contraction.

Case 2 $\Delta \leq U/\alpha$. Then $\mathcal{A}$ achieves a cost reduction of at least

$$h\left(\frac{\alpha\Delta}{U}\right) = \frac{\alpha^2\Delta^2}{2U} = \gamma\Delta^2 \ \ ,$$

where

$$\gamma := \frac{\alpha^2}{2U} \ \ . \tag{12}$$

Thus, the difference $\Delta = f(x) - f(x^*)$ will shrink after the next iteration to

$$\Delta - \gamma\Delta^2$$

or to a smaller value.

Recall from (2) that the sequence $(\Delta_n)_{n\geq 0}$ keeps track of the difference between the value of the current feasible solution and the value of an optimal solution. In view of the two cases described above, our pessimistic analysis obviously runs through two phases:

Phase 1 As long as $\Delta_n > U/\alpha$, we calculate with a cost reduction of at least $\alpha\Delta_n/2$.

Phase 2 As soon as $\Delta_n \leq U/\alpha$, we calculate with a cost reduction of at least $\gamma\Delta_n^2$.

Let us first assume that $\varepsilon < U/\alpha$ (and postpone the case $\varepsilon \geq U/\alpha$ to the end of this subsection). The number of iterations in phase 1 is not larger than the smallest $n_0 \geq 0$ that satisfies the second inequality in

$$\Delta_0\left(1 - \frac{\alpha}{2}\right)^{n_0} < \Delta_0 e^{-n_0\alpha/2} \leq \varepsilon < \frac{U}{\alpha} \ \ ,$$

that is

$$n_0 := \max\left\{0, \left\lceil \frac{2}{\alpha}\ln\left(\frac{\Delta_0}{\varepsilon}\right)\right\rceil\right\} \ \ . \tag{13}$$

In phase 2, $(\Delta_n)_{n\geq n_0}$ evolves according to

$$\Delta_{n+1} \leq \Delta_n - \gamma\Delta_n^2 = \Delta_n(1 - \gamma\Delta_n) \ \ . \tag{14}$$

Recall that $\Delta_i = f(x^i) - f(x^*) \geq 0$ for every i. As for the iterations considered in phase 2 within our analysis, we can make the stronger (pessimistic) assumption $\Delta_i > 0$.[10] Following Dunn (1979), we can therefore consider the reciprocals $\delta_n := 1/\Delta_n$. Note that (14) and $\Delta_{n+1} > 0$ imply that $0 \leq \gamma\Delta_n < 1$ and so for each $n \geq n_0$ the following relation holds:

$$\delta_{n+1} - \delta_n \geq \frac{1}{\Delta_n(1-\gamma\Delta_n)} - \frac{1}{\Delta_n} = \frac{\gamma}{1-\gamma\Delta_n} \geq \gamma \ .$$

Therefore

$$\delta_n = \delta_{n_0} + \sum_{j=n_0}^{n-1} (\delta_{j+1} - \delta_j) \geq \gamma(n-n_0)$$

and consequently

$$\Delta_n = \frac{1}{\delta_n} \leq \frac{1}{\gamma(n-n_0)} \ .$$

It follows that $\Delta_n \leq \varepsilon$ after

$$n - n_0 := \left\lceil \frac{1}{\gamma\varepsilon} \right\rceil$$

iterations in phase 2. Thus the total number of iterations in both phases needed to be within ε of optimality is bounded by

$$n_0 + \left\lceil \frac{1}{\gamma\varepsilon} \right\rceil \overset{(12),(13)}{\leq} \left\lceil \frac{2U}{\alpha^2\varepsilon} \right\rceil + \max\left\{0, \left\lceil \frac{2}{\alpha}\ln\left(\frac{\Delta_0}{\varepsilon}\right) \right\rceil \right\}$$

iterations. Plugging in the definition of U from (11), we obtain (8).
Let us now finally discuss the case $U = qL_{max}S_{max}^2 \leq \varepsilon\alpha$. Since $\varepsilon \geq U/\alpha$, we come within ε of optimality during phase 1. The number of iterations required for this is the smallest n_0 that satisfies

$$\Delta_0 \left(1 - \frac{\alpha}{2}\right)^{n_0} < \Delta_0 e^{-n_0\alpha/2} \leq \varepsilon \ .$$

Thus, we are within ε of optimality after

$$\max\left\{0, \left\lceil \frac{2}{\alpha}\ln\left(\frac{\Delta_0}{\varepsilon}\right) \right\rceil \right\}$$

iterations. This completes the proof of the first statement in Theorem 4.

3.3 Proof of the 2nd Statement in Theorem 4

Before we present the proof, we recall some definitions and facts from the theory of Linear Programming.[11]

An *instance of Linear Programming* is given by a linear target function $c^\top x$ that has to be minimized (or maximized) subject to some linear equality and inequality constraints. A vector that satisfies the constraints is called a *feasible solution*. The instance is called *feasible* if it has a feasible solution. It is called *bounded* if it has a finite optimal value. An instance of the form

$$\max_x c^\top x \text{ s.t. } Ax = b \ , \ x \geq 0 \ ,$$

10. Otherwise phase 2 ends with an optimal solution even earlier.

11. See any standard text book about Combinatorial Optimization (for example Papadimitriou and Steiglitz, 1982).

where $b,c \in \mathbb{R}^n$ and $A \in \mathbb{R}^{m\times n}$, $m,n \in \mathbb{N}$, is said to be in *standard form*. We can assume that A has rank m (after removal of redundant equations if necessary). An $(m\times m)$-sub-matrix B of A formed by m linearly independent columns of A, say $A_{i_1},\dots,A_{i_m}$, $1 \le i_1 < \cdots < i_m \le n$, is called a *basis*. The *basic solution corresponding to B* is obtained by setting $x_i = 0$ for every $i \notin \{i_1,\dots,i_m\}$ and by setting x_{i_l} equal to the l'th component of $B^{-1}b$ for $l = 1,\dots,m$. Note that any basic solution (corresponding to some basis) has at most $m = \text{rank}(A)$ many nonzero components. If a basic solution happens to be feasible, it is called a *basic feasible solution*. It is well-known that, for every feasible and bounded instance of Linear Programming in standard form, there exists a basic feasible solution that is optimal (among all feasible solutions).

We are now prepared for the proof of the second statement in Theorem 4. We start with a short outline. Let x be a feasible but sub-optimal solution for $\mathcal{P}$. We have to show that one can efficiently construct a working set I such that $|I| \le k+1$ and

$$\sigma(x|I) \ge \frac{1}{m}\sigma(x) \ . \tag{15}$$

To this end, we will proceed as follows. We consider auxiliary instances $\mathcal{P}_x, \mathcal{P}'_x, \mathcal{P}''_x$ of Linear Programming (denoted as LP in the sequel). The optimal values for $\mathcal{P}_x$ and $\mathcal{P}'_x$, are both shown to coincide with $\sigma(x)$. From $\mathcal{P}'_x$, we derive (basically by aggregating several equality constraints into a single-one) the instance $\mathcal{P}''_x$, whose optimal basic solution will represent a working set I of size at most $k+1$. A comparison of the three problem instances will finally reveal that I satisfies (15).

We now move on to the technical implementation of this plan. Recall from (5) that

$$\sigma(x) = \sup_{x' \in R(\mathcal{P})} \nabla f(x)^\top (x - x') = \nabla f(x)^\top d \ ,$$

where $d = x - x'$. Thus $\sigma(x)$ is the optimal value of the following instance $\mathcal{P}_x$ of LP:

$$\max_d \nabla f(x)^\top d \text{ s.t. } Ad = \vec{0}, l \le x - d \le r \ .$$

We set

$$\mu^+ := x - l \text{ and } \mu^- := r - x$$

and split d into positive and non-positive components d^+ and d^- respectively:

$$d = d^+ - d^-,\ d^+, d^- \ge \vec{0},\ \forall i = 1,\dots,m : d_i^+ d_i^- = 0 \ . \tag{16}$$

With these notations, $\sigma(x)$ also coincides with the optimal value of the following instance $\mathcal{P}'_x$ of LP:

$$\max_{d^+,d^-} \begin{pmatrix} \nabla f(x) \\ -\nabla f(x) \end{pmatrix}^\top \begin{pmatrix} d^+ \\ d^- \end{pmatrix}$$

subject to

$$\begin{array}{rcl} [A, -A] \begin{pmatrix} d^+ \\ d^- \end{pmatrix} & = & \vec{0}, \\ \vec{0} \le d^+ \le \mu^+ & , & \vec{0} \le d^- \le \mu^- . \end{array}$$

The third instance $\mathcal{P}''_x$ of LP that we consider has an additional slack variable ξ and is given as follows:

$$\max_{d^+,d^-,\xi} \begin{pmatrix} \nabla f(x) \\ -\nabla f(x) \end{pmatrix}^\top \begin{pmatrix} d^+ \\ d^- \end{pmatrix}$$

subject to

$$\begin{aligned} \forall i=1,\ldots,m: \mu_i^- = 0 \Rightarrow d_i^- = 0 \quad &, \quad \mu_i^+ = 0 \Rightarrow d_i^+ = 0, \qquad (17)\\ [A,-A]\begin{pmatrix} d^+ \\ d^- \end{pmatrix} &= \vec{0}, \\ \sum_{i:\mu_i^+>0} \frac{1}{\mu_i^+} d_i^+ + \sum_{i:\mu_i^->0} \frac{1}{\mu_i^-} d_i^- + \xi &= 1, \\ d^+, d^- \geq \vec{0} \quad &, \quad \xi \geq 0. \end{aligned}$$

We briefly note that we do not have to count the equality constraints in (17) because we may simply remove the variables that are set to zero from the problem instance. Recall that matrix A represents k equality constraints. Thus, $\mathcal{P}''_x$ is a linear program in standard form with $k+1$ equality constraints. Its optimal basic feasible solution has therefore at most $k+1$ non-zero components. The following observations (where d, d^+, d^- are related according to (16)) are easy to verify:

1. If $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is a feasible solution for $\mathcal{P}'_x$ with value p, then $\frac{1}{m}\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is a feasible solution for $\mathcal{P}''_x$ with value p/m.

2. If $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is a feasible solution for $\mathcal{P}''_x$ with value p, then $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is also a feasible solution for $\mathcal{P}'_x$ with value p.

Recall that $\sigma(x)$ is the value of an optimal solution for $\mathcal{P}'_x$. We may conclude from our observations that the value of an optimal solution for $\mathcal{P}''_x$, say $\sigma'(x)$, satisfies $\sigma'(x) \geq \sigma(x)/m$. Now consider an optimal basic feasible solution $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ for $\mathcal{P}''_x$ with value $\sigma'(x)$. Let

$$I := \{i \in \{1,\ldots,m\} |\ d_i^+ \neq 0 \text{ or } d_i^- \neq 0\} \ .$$

Clearly, $|I| \leq k+1$. Since $d = d^+ - d^-$ is a feasible solution for $\mathcal{P}$ (still with value $\sigma'(x)$) that differs from x only in coordinates from I, we may conclude that

$$\sigma(x|I) \geq \sigma'(x) \geq \frac{1}{m}\sigma(x) \ .$$

In other words, working set I satisfies (15). The time required to compute I is dominated by the time required to solve the linear program $\mathcal{P}''_x$. This can be done in polynomial time. Since a linear program with a constant number of variables (or a linear program in standard form with a constant number of equality constraints[12]) can be solved in linear time (Megiddo, 1984), the proof for the 2nd statement of Theorem 4 is completed.

12. That means the dual linear program has constant number of variables.

3.4 Relation between Certifying Sets and Sparse Witnesses of Sub-optimality

Before we come to the main issue of this section, we briefly recall some well-known definitions and facts from the theory of Linear Programming (see Papadimitriou and Steiglitz, 1982, Definition 3.1 and Theorems 3.1 and 3.2).

For every (primal) instance of Linear Programming, one can construct the so-called *dual instance* according to the following scheme:

primal		dual
$\min_x c^\top x$		$\max_\lambda \lambda^\top b$
$a_i^\top x = b_i$	$i \in M$	—
$a_i^\top x \geq b_i$	$i \in \bar{M}$	$\lambda_i \geq 0$
$x_j \geq 0$	$j \in N$	$\lambda^\top A_j \leq c_j$
—	$j \in \bar{N}$	$\lambda^\top A_j = c_j$

In this scheme, $A \in \mathbb{R}^{m\times n}$ is a matrix with column vectors $A_1,\dots,A_n$ and row vectors $a_1^\top,\dots,a_m^\top$, $N \subseteq \{1,\dots,n\}$ is the set of indices corresponding to primal variables that are constrained to be non-negative, and $M \subseteq \{1,\dots,m\}$ is the set of indices corresponding to equality constraints (as opposed to inequality constraints) in the primal. It is well known that the optimal values of the primal and the dual coincide (assuming that the optimal values exist). Moreover, the dual of the dual coincides with the primal.

Let us now return to the main issue of this section. Consider again the function $\sigma(x|I)$ from (6) and the function $C_I(x)$ from (4). Here comes the main result of this section:

Lemma 5 $\sigma(x|I) = C_I(x)$.

Proof Given a subset $I \subset \{1,\dots,m\}$, $|I| = q$, and a vector $x \in \mathbb{R}^m$, we will write shortly $x_I \in \mathbb{R}^q$ for the vector consisting only of entries $x_i, i \in I$. Recall that $A_i \in \mathbb{R}^k$ denotes the i-th column of matrix A and write $A_I \in \mathbb{R}^{k\times q}$ for the matrix consisting of columns A_i such that $i \in I$.
An inspection of (6) reveals that $\sigma(x|I)$ can equivalently be written as follows:

$$\sigma(x|I) = \sup_{d\in\mathbb{R}^q} \left\{ \nabla f(x)_I^\top d \mid A_I d = 0,\ l_I \leq x_I - d \leq r_I \right\} \ .$$

This is an (obviously feasible and bounded) instance of Linear Programming with the following inequality constraints

$$\begin{pmatrix} \mathrm{Id}_q \\ -\mathrm{Id}_q \end{pmatrix} d \leq \begin{pmatrix} x_I - l_I \\ r_I - x_I \end{pmatrix} \ ,$$

where $\mathrm{Id}_q \in \mathbb{R}^{q\times q}$ denotes the identity-matrix. According to LP-duality, $\sigma(x|I)$ can also be viewed as the minimum cost of the dual minimization problem:

$$\sigma(x|I) = \inf_{\substack{\alpha,\beta\in\mathbb{R}^q \\ \lambda\in\mathbb{R}^k}} \left\{ \begin{pmatrix} \alpha \\ \beta \end{pmatrix}^\top \begin{pmatrix} x_I - l_I \\ r_I - x_I \end{pmatrix} \middle|\ A_I^\top \lambda + \alpha - \beta = \nabla f(x)_I,\ \alpha,\beta \geq 0 \right\} \ . \tag{18}$$

It is easy to see that the optimal choice of α and β is as follows:

$$\alpha_i = \max\{0, \nabla f(x)_i - A_i^\top \lambda\} \qquad \text{and} \qquad \beta_i = \max\{0, A_i^\top \lambda - \nabla f(x)_i\} \ .$$

Plugging these settings into (18), we finally get

$$\begin{aligned}
&\sigma(x|I) = \\
&\quad \inf_{\lambda \in \mathbb{R}^k} \sum_{i \in I} \Big((x_i - l_i) \max\{0, \nabla f(x)_i - A_i^\top \lambda\} + (r_i - x_i) \max\{0, A_i^\top \lambda - \nabla f(x)_i\} \Big) \\
&\quad = C_I(x) \enspace .
\end{aligned}$$

■

Lemma 5 is remarkable because the coincidence of the two functions was not known before and their definitions originated from different motivations:

- $C_I(x)$ was designed such as to satisfy the axioms for sparse witnesses of sub-optimality.
- $\sigma(x|I)$ was designed as a generalization of the corresponding function for rate certifying pairs.

Definition 6 (WSS) *The problem "Working Set Selection for CQO", denoted briefly as WSS(CQO), is the following computational problem: given an instance of CQO (see Definition 1) augmented by a feasible solution x and an upper bound $q \geq k+1$ on the size of the working set, find a working set $I_* \subseteq \{1, \ldots, n\}$ that maximizes $\sigma(x|I)$ subject to $|I| \leq q$.*
An approximation algorithm for WSS(CQO) is called ρ-optimal[13] *if, for every instance of WSS(CQO), it finds a working set I_0 such that $|I_0| \leq q$ and $\sigma(x|I_0) \geq \rho \max_{I:|I| \leq q} \sigma(x|I)$.*
The corresponding notions for the subproblem WSS(SVO) are defined analogously.

Rate certifying algorithms and approximation algorithms for WSS(CQO) are closely related:

Corollary 7 *The following holds for WSS(CQO):*[14]

1. *A strong $(\vec{\alpha}, k+1)$-rate certifying algorithm can be viewed as an $\vec{\alpha}(m)$-optimal approximation algorithm for WSS(CQO).*
2. *An approximation algorithm for WSS(CQO) with performance ratio ρ can be viewed as a strong $(\vec{\alpha}, k+1)$-rate certifying algorithm where $\vec{\alpha}(m) = \rho/m$.*

Proof

1. A strong $(\vec{\alpha}, k+1)$-rate certifying algorithm applied to an instance of WSS(CQO) constructs a working set I_0 of size at most $k+1$ such that $\sigma(x|I_0) \geq \vec{\alpha}(m)\sigma(x)$. Clearly, $\sigma(x) \geq \max_{I:|I| \leq q} \sigma(x|I)$. Thus, $\sigma(x|I_0)$ matches the optimal value up to factor $\vec{\alpha}(m)$.
2. An approximation algorithm for WSS(CQO) can be used as the procedure within a decomposition algorithm that performs the working set selection. At every iteration, it is applied to an instance of WSS(CQO) where $q = k+1$. It then constructs a working set I_0 such that $|I_0| \leq q$ and $\sigma(x|I_0) \geq \rho \max_{I:|I| \leq k+1} \sigma(x|I)$. We know from Theorem 4 that $\max_{I:|I| \leq k+1} \sigma(x|I) \geq \sigma(x)/m$. Thus $\sigma(x|I_0)$ matches the optimal value up to factor ρ/m.

13. Here, ρ may be a function in the input parameters.
14. The analogous statements for WSS(SVO) hold as well.

■

Corollary 7 immediately implies that the strong $(1/m, k+1)$-rate certifying algorithm described in Section 3.3 can be viewed as a $1/m$-optimal approximation algorithm for WSS(CQO). This result can however be strengthened:

Corollary 8 *The strong $(1/m, k+1)$-rate certifying algorithm described in Section 3.3 can be viewed as a $1/q$-optimal approximation algorithm for WSS(CQO).*

Proof We use again the notations (like, for example, $\mathcal{P}'_x, \mathcal{P}''_x$) that were introduced in Section 3.3. Every instance of WSS(CQO) can be cast as an instance $\mathcal{P}'_{x,q}$ resulting from an instance $\mathcal{P}'_x$ by adding the constraint that the total number of non-zero components in d^+, d^- is bounded by q. The algorithm from Section 3.3 returns an optimal solution for the instance $\mathcal{P}''_x$. The corollary is now an immediate consequence of the following observations:

1. If $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is a feasible solution for $\mathcal{P}'_{x,q}$ with value p, then $\frac{1}{q}\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is a feasible solution for $\mathcal{P}''_x$ with value p/q.
2. If $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is a basic feasible solution for $\mathcal{P}''_x$ with value p, then $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is also a feasible solution for $\mathcal{P}'_{x,q}$ with value p.

■

Our results nicely strengthen (or complement) what was known before about working set selection w.r.t. $C_I(x)$:

- For general $q-$sparse witnesses $\mathcal{C}_I(x)$ List and Simon (2004) have shown that the "strict rule" of always selecting a working set I_* that maximizes $\mathcal{C}_I(x)$ subject to $|I| \leq q$ leads to cost-optimal feasible solutions in the limit (but no convergence rate was provided). Simon (2004) has shown a similar result for the "relaxed rule". For the special case $\mathcal{C}_I(x) = C_I(x)$ Theorem 4 extends these mere convergence results by providing a convergence rate, since $C_I(x) = \sigma(x|I)$.
- Simon (2004) has shown that (a canonical combinatorial abstraction of) WSS(SVO) is NP-complete, but admits a full polynomial time approximation scheme. According to Corollary 8, the algorithm described in Section 3.3 is an approximation algorithm with performance ratio $1/m$ for the more general problem WSS(CQO).

4. Conclusions and Open Problems

We have presented an analysis of a decomposition algorithm that leads to the to-date strongest theoretical performance guarantees within the "rate certifying pair" approach. Our analysis holds uniformly for any instance of Convex Quadratic Optimization (with box-constraints) and certainly covers most of the variants of SVM-optimization. Our results can be seen as an extension of (and an improvement on) earlier work by Hush and Scovel (2003) about rate certifying pairs, and by List and Simon (2004) and Simon (2004) about sparse witnesses of sub-optimality.

As explained in the introduction already, there are competing approaches like, for example, the approach based on maximally KKT-violating pairs or approaches based on an iterative inclusion of support vectors. Recent experiments by Hush et al. (2006) indicate that

- the approach based on maximally KKT-violating pairs converges faster than the approach based on rate certifying pairs,
- but a hybrid approach that makes a greedy choice among the maximally KKT-violating pair and pairs that come up in the rate certifying pair approach is often superior.

At the time being, these experimental results are not supported by theoretical analysis. We leave the task of providing a theoretical explanation for the empirically good performance of the hybrid approach (or even the original approach based on KKT-violating pairs) as an object of future research.

Acknowledgments

This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views. This work was furthermore supported by the Deutsche Forschungsgemeinschaft Grant SI 498/7-1.

References

Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152, 1992.

Steven Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Chih-Chung Chang, Chih-Wei Hsu, and Chih-Jen Lin. The analysis of decomposition methods for support vector machines. *IEEE Transactions on Neural Networks*, 11(4):248–250, 2000.

Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Available from http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

Pai-Hsuen Chen, Rong-En Fan, and Chih-Jen Lin. Training support vector machines via SMO-type decomposition methods. In *Proceedings of the 16th International Conference on Algorithmic Learning Theory*, pages 45–62, 2005.

Pai-Hsuen Chen, Rong-En Fan, and Chih-Jen Lin. A study on SMO-type decomposition methods for support vector machines. *IEEE Transactions on Neural Networks*, 17(4):893–908, 2006.

J. Dunn. Rates of convergence for conditional gradient algorithms near singular and non-singular extremals. *SIAM J. Control and Optimization*, 17(2):187–211, 1979.

Chih-Wei Hsu and Chih-Jen Lin. A simple decomposition method for support vector machines. *Machine Learning*, 46(1–3):291–314, 2002.

Don Hush, Patrick Kelly, Clint Scovel, and Ingo Steinwart. QP Algorithms with Guaranteed Aaccuracy and Run Time for Support Vector Machines. *Journal of Machine Learning Research*, 7: 733–769, May 2006.

Don Hush and Clint Scovel. Polynomial-time decomposition algorithms for support vector machines. *Machine Learning*, 51(1):51–71, 2003.

Thorsten Joachims. Making large scale SVM learning practical. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods—Support Vector Learning*, pages 169–184. MIT Press, 1998.

S. Sathiya Keerthi and E. G. Gilbert. Convergence of a generalized SMO algorithm for SVM classifier design. *Machine Learning*, 46(1–3):351–360, 2002.

S. Sathiya Keerthi, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and K. R. K. Murthy. Improvements to SMO algorithm for SVM regression. *IEEE Transactions on Neural Networks*, 11 (5):1188–1193, 2000.

S. Sathiya Keerthi, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and K. R. K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation*, 13(3): 637–649, 2001.

Pavel Laskov. Feasible direction decomposition algorithms for training support vector machines. *Machine Learning*, 46(1–3):315–349, 2002.

Shuo-Peng Liao, Hsuan-Tien Lin, and Chih-Jen Lin. A note on the decomposition methods for support vector regression. *Neural Computation*, 14(6):1267–1281, 2002.

Chih-Jen Lin. Linear convergence of a decomposition method for support vector machines, 2001a. Available from http://www.csie.ntu.edu.tw/∼cjlin/papers/linearconv.pdf.

Chih-Jen Lin. On the convergence of the decomposition method for support vector machines. *IEEE Transactions on Neural Networks*, 12(6):1288–1298, 2001b.

Chih-Jen Lin. Asymptotic convergence of an SMO algorithm without any assumptions. *IEEE Transactions on Neural Networks*, 13(1):248–250, 2002a.

Chih-Jen Lin. A formal analysis of stopping criteria of decomposition methods for support vector machines. *IEEE Transactions on Neural Networks*, 13(5):1045–1052, 2002b.

Nikolas List. Convergence of a generalized gradient selection approach for the decomposition method. In *Proceedings of the 15th International Conference on Algorithmic Learning Theory*, pages 338–349, 2004.

Nikolas List and Hans Ulrich Simon. A general convergence theorem for the decomposition method. In *Proceedings of the 17th Annual Conference on Computational Learning Theory*, pages 363–377, 2004.

Olvi L. Mangasarian and David R. Musicant. Successive overrelaxation for support vector machines. *IEEE Transactions on Neural Networks*, 10(5):1032–1037, 1999.

Olvi L. Mangasarian and David R. Musicant. Active support vector machine classification. In *Advances in Neural Information Processing Systems 12*, pages 577–583. MIT Press, 2000.

Olvi L. Mangasarian and David R. Musicant. Lagrangian support vector machines. *Journal of Machine Learning Research*, 1:161–177, 2001.

Nimrod Megiddo. Linear programming in linear time when the dimension is fixed. *Journal of the Association on Computing Machinery*, 31(1):114–127, 1984.

Edgar E. Osuna, Robert Freund, and Federico Girosi. Training support vector machines: an application to face detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 130–136, 1997.

Christos H. Papadimitriou and Kenneth Steiglitz *Combinatorial Optimization*. Prentice Hall, 1982.

John C. Platt. Fast training of support vector machines using sequential minimal optimization. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods—Support Vector Learning*, pages 185–208. MIT Press, 1998.

Craig Saunders, Mark O. Stitson, Jason Weston, Leon Bottou, Bernhard Schölkopf, and Alexander J. Smola. Support vector machine reference manual. Technical Report CSD-TR-98-03, Royal Holloway, University of London, Egham, UK, 1998.

Hans Ulrich Simon. On the complexity of working set selection. In *Proceedings of the 15th International Conference on Algorithmic Learning Theory*, pages 324–337, 2004. A full version of the paper will appear in TCS.

Vladimir Vapnik. *Statistical Learning Theory*. Wiley Series on Adaptive and Learning Systems for Signal Processing, Communications, and Control. John Wiley & Sons, 1998.

S. V. N. Vishwanthan, Alexander J. Smola, and M. Narasimha Murty. SimpleSVM. In *Proceedings of the 20th International Conference on Machine Learning*, pages 760–767, 2003.

Journal of Machine Learning Research 8 (2007) 323-360 Submitted 9/05; Revised 7/06; Published 2/07

Dynamics and Generalization Ability of LVQ Algorithms

Michael Biehl M.BIEHL@RUG.NL
Anarta Ghosh ANARTA@GMAIL.COM
Institute for Mathematics and Computing Science
University of Groningen
P.O. Box 800, NL-9700 AV Groningen, The Netherlands

Barbara Hammer HAMMER@IN.TU-CLAUSTHAL.DE
Institute of Computer Science
Clausthal University of Technology
D-38678 Clausthal-Zellerfeld, Germany

Editor: Yoshua Bengio

Abstract

Learning vector quantization (LVQ) schemes constitute intuitive, powerful classification heuristics with numerous successful applications but, so far, limited theoretical background. We study LVQ rigorously within a simplifying model situation: two competing prototypes are trained from a sequence of examples drawn from a mixture of Gaussians. Concepts from statistical physics and the theory of on-line learning allow for an exact description of the training dynamics in high-dimensional feature space. The analysis yields typical learning curves, convergence properties, and achievable generalization abilities. This is also possible for heuristic training schemes which do not relate to a cost function. We compare the performance of several algorithms, including Kohonen's LVQ1 and LVQ+/-, a limiting case of LVQ2.1. The former shows close to optimal performance, while LVQ+/- displays divergent behavior. We investigate how early stopping can overcome this difficulty. Furthermore, we study a crisp version of robust soft LVQ, which was recently derived from a statistical formulation. Surprisingly, it exhibits relatively poor generalization. Performance improves if a window for the selection of data is introduced; the resulting algorithm corresponds to cost function based LVQ2. The dependence of these results on the model parameters, for example, prior class probabilities, is investigated systematically, simulations confirm our analytical findings.

Keywords: prototype based classification, learning vector quantization, Winner-Takes-All algorithms, on-line learning, competitive learning

1. Introduction

The term *learning vector quantization* (LVQ) has been coined for a family of algorithms which is widely used in the classification of potentially high-dimensional data. Successful applications of LVQ include such diverse problems like medical image or data analysis, fault detection in technical systems, or the classification of satellite spectral data (Bojer et al., 2003; Kuncheva, 2004; Schleif et al., 2006; Villmann et al., 2003). An up to date overview and extensive bibliography is available at a repository which is maintained by the Neural Networks Research Centre, Helsinki (2002).

The popularity of the approach is due to several attractive features: LVQ procedures are easy to implement and intuitively clear. This makes them particularly interesting for researchers and practitioners outside the machine learning community who are searching for robust classification schemes without the black-box character of many neural methods. The classification of LVQ is based on a distance measure, usually the Euclidean distance, which quantifies the similarity of given data with so-called prototype or codebook vectors representing the classes. The prototypes are determined in a training process from labeled example data and can be interpreted in a straightforward way as they capture essential features of the data in the very same space. This is in contrast with, say, adaptive weights in feedforward neural networks or support vector machines which do not allow for immediate interpretation as easily since they are embedded in a different space or at atypical borderline positions of the data instead of typical regions. Other very attractive features of LVQ are the natural way in which it can be applied to multi-class problems and the fact that the complexity of LVQ networks can be controlled during training according to the specific needs.

In general, several prototypes can be employed to represent one of the classes. In simple *hard* or *crisp* schemes, a data or feature vector is assigned to the closest of all prototypes and the corresponding class. Extensions of this deterministic scheme to probabilistic *soft* assignments and classification are straightforward but will not be considered here. Plausible training prescriptions exist which mostly employ the concept of competitive learning and adaptation by means of Hebbian terms. Prototypes are updated according to their distance from given example data. Thereby, given a training pattern, the closest prototype vector or a set of closest vectors, the so-called *winners* are identified. These vectors are then moved towards (away from) the data if their class label coincides with (differs from) that of the example, respectively.

Both, training algorithms and the resulting classification schemes are fairly easy to implement and fast. In practice, the computational effort of LVQ training usually scales linearly with the training set size, and that of classification depends only on the (fixed) number of prototypes and the input dimensionality. Furthermore, training is incremental, such that the adaptation of a classifier to novel data which becomes available after a first training phase is straightforward. Despite this simplicity, LVQ is very powerful since every separating hypercurve can, in principle, be approximated by LVQ networks with sufficiently many prototypes. Furthermore, recent variations of LVQ allow for the incorporation of problem specific metrics or kernels which can be adapted during training such that very few prototypes can model quite complex behavior (Hammer et al., 2005b,c).

However, the theoretical understanding of the convergence properties, stability, and achievable generalization ability of many LVQ algorithms appears fairly limited. Many LVQ procedures, including Kohonen's original formulation (LVQ1) are based on heuristic arguments. A large variety of modifications of the basic scheme have been suggested which aim at larger flexibility, faster convergence or better approximation of Bayesian decision boundaries, such as LVQ2.1, LVQ3, OLVQ, RSLVQ, or GLVQ (Kohonen, 1997, 1990; Pregenzer et al., 1996; Sato and Yamada, 1995). Clearly, the ultimate goal of training is good generalization, that is, the correct classification of novel data with high probability after training. However, the above mentioned algorithms differ considerably in their learning dynamics, stability, and generalization performance, while the theoretical background of this behavior is hardly understood.

Recently, a few attempts to put LVQ type algorithms on solid mathematical grounds have been made. Remarkably, LVQ-type classifiers fall into the class of *large-margin algorithms* which allow for dimensionality independent generalization bounds, as pointed out first by Crammer et al. (2003). Here, the term margin refers to the so-called hypothesis margin of an LVQ classifier: the distance the classification boundary, that is, the respective prototypes, can be moved without changing the classification of the given data points. Similar bounds can also be derived for recent variants of LVQ which substitute the standard Euclidean metric by an alternative adaptive choice involving relevance factors (Hammer et al., 2005a; Hammer and Villmann, 2002). Thus, LVQ type networks seem promising candidates for the classification of high-dimensional data sets.

However, standard LVQ training does not directly aim at an optimization of the margin or even an optimization of the classification error, usually. Rather, the learning algorithm is often purely heuristically motivated and does not directly follow any learning objective or explicit cost function. Apart from the fact that the quality of generalization in these methods is therefore not clear, often dynamical problems such as divergence of the algorithms can be observed. An important example is LVQ2.1 and similar strategies which can display divergent behavior unless a proper (also heuristic) window rule is included in the update scheme. Few approaches try to cope with this problem by explicitly grounding the update scheme on a cost function (see, for example, Bottou, 1991). A prominent example has been proposed by Sato and Yamada (1995) together with a discussion of the stability (Sato and Yamada, 1998), however, without considering the borders of receptive fields. A generalization which allows the incorporation of more general metrics, accompanied by an investigation of the borders, has been presented by Hammer and Villmann (2002) and Hammer et al. (2005c). Recently, two models which provide a cost function by means of soft assignments have been proposed by Seo et al. (2003) and Seo and Obermayer (2003). While the first method does not possess a crisp limit, the second one does. However, only the cost function discussed by Sato and Yamada (1995) and Hammer and Villmann (2002) is directly connected to the hypothesis margin, as pointed out by Hammer et al. (2005a). For all these cost functions, the connection to the classification error is not obvious. Thus, there is a clear need for a rigorous analysis of LVQ type learning schemes which allows for a systematic judgement of their dynamical properties and generalization ability.

In this work we discuss and employ a theoretical framework which makes possible the systematic investigation and comparison of LVQ training procedures. The analysis is performed for model situations in which training uses a sequence of uncorrelated, randomized example data. In the theory of such on-line learning processes, concepts known from statistical physics have been applied with great success in the context of supervised and unsupervised learning. This includes perceptron training, gradient based training of layered feedforward neural networks, and unsupervised clustering or principal component analysis (see Watkin et al., 1993; Biehl and Caticha, 2003; Saad, 1999; Engel and van den Broeck, 2001, for additional references). Here, we transfer these methods to LVQ type learning, and we are capable of presenting quite unexpected insights into the dynamical behavior of several important learning schemes.

The essential ingredients of the approach are (1) the consideration of very large systems in the so-called thermodynamic limit and (2) the performing of averages over the randomness or *disorder* contained in the data. A small number of characteristic quantities is sufficient to describe the essential properties of very large systems. Under simplifying assumptions, the evolution of these macroscopic *order parameters* is given by a system of deterministic coupled ordinary differential equations (ODE) which describe the dynamics of on-line learning exactly in the thermodynamic limit.

The formalism enables us to calculate, for instance, typical learning curves exactly within the framework of model situations. We compare the generalization behavior of several LVQ algorithms. Their asymptotic behavior in the limit of infinitely many examples is of particular relevance in this context. We evaluate the achievable generalization error as a function of the prior frequencies of classes and other model parameters. Thus, for the considered model situation, we can rigorously compare the performance of LVQ-type learning schemes and identify the best training method.

The paper is organized as follows: In the next section we introduce the model data and the structure of the LVQ classifier. We also describe the considered training algorithms which include Kohonen's LVQ1 and a limit version of LVQ2.1 or LVQ-LR. We will use the term LVQ+/- for this prescription and we consider it with and without an idealized *early stopping* procedure. Furthermore we study a *crisp* version of RSLVQ (Seo and Obermayer, 2003), which will be referred to as learning from mistakes (LFM). Finally, we consider LFM-W, a variant of the latter which selects data from a *window* close to the decision boundary and relates to the cost function based LVQ2 algorithm (Kohonen et al., 1988; Bottou, 1991). In Section 3 we outline the mathematical treatment. The formalism and its application to the above mentioned algorithms are detailed in Appendix A. Relevant features of the learning curves, including the achievable generalization ability are discussed and compared for the different training prescriptions in Section 4. Finally, we summarize our findings in Section 5 and conclude with a discussion of prospective projects and extensions of our studies.

Some aspects of this work have been communicated previously, in much lesser detail, at the *Workshop on the Self-Organizing Map* in Paris, 2005 (Ghosh et al., 2005).

2. The Model

In this paper we investigate the dynamics of different LVQ algorithms in the framework of a simplifying model situation: High-dimensional data from a mixture of two overlapping Gaussian clusters are presented to a system of two prototype vectors each of which represents one of the classes. Whereas the model appears very much simplified in comparison with real world multi-class, multi-prototype problems, it provides a setting in which it is well possible to study unexpected, essential, and non-trivial features of the learning scenarios.

2.1 Prototype Vectors

We will investigate situations with only two prototype or codebook vectors $w_S \in \mathbb{R}^N$. Here, the subscript $S = \pm 1$ indicates which class of data the vector is supposed to represent. For the sake of brevity we will frequently use the shorthand subscripts $+$ or $-$ for $+1$ and -1, respectively. The classification as parameterized by the prototypes is based on Euclidean distances: Any given input $\xi \in \mathbb{R}^N$ will be assigned to the class of the closest prototype. In our model situation, the classification result is S whenever $\mid w_{+S} - \xi \mid < \mid w_{-S} - \xi \mid$.

Note that, in principle, the simple LVQ classifier with only two prototypes could be replaced by a linearly separable classification $S = \text{sign}[w_{perc} \cdot \xi - \theta_{perc}]$ with *perceptron* weight vector $w_{perc} = (w_+ - w_-)$ and threshold $\theta_{perc} = \left(w_+^2 - w_-^2\right)/2$. Here, however, we are interested in the more complex learning dynamics of independent codebook vectors as provided by LVQ algorithms. We will demonstrate that our simple example scenario already displays highly non-trivial features and provides valuable insights in the more general training of several prototypes.

2.2 The Data

Throughout the following we consider random inputs which are generated according to a bimodal distribution. We assume that vectors $\xi \in \mathbb{R}^N$ are drawn independently according to the density

$$P(\xi) = \sum_{\sigma=\pm 1} p_\sigma P(\xi \mid \sigma) \;\; \text{with} \;\; P(\xi \mid \sigma) = \frac{1}{(2\pi v_\sigma)^{N/2}} \exp\left[-\frac{1}{2 v_\sigma}\left(\xi - \lambda B_\sigma\right)^2\right]. \tag{1}$$

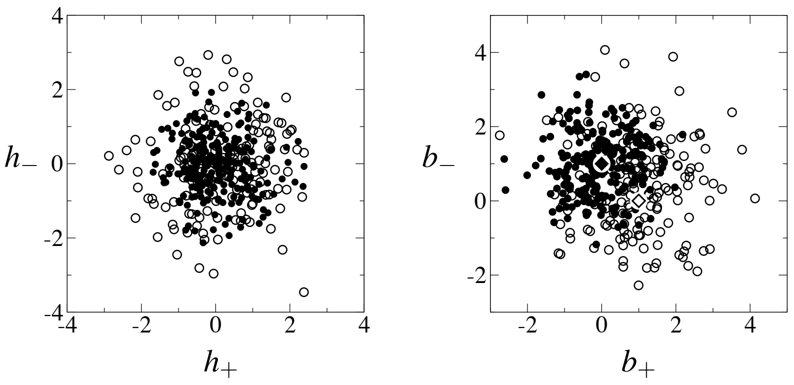


Figure 1: Data as generated according to the density (1) in $N = 200$ dimensions with prior weights $p_- = 0.6, p_+ = 0.4$ and variances $v_- = 0.64, v_+ = 1.44$. Open (filled) circles represent 160 (240) vectors ξ from clusters centered about orthonormal vectors λB_+ (λB_-) with $\lambda = 1$, respectively. The left panel shows the projections $h_\pm = w_\pm \cdot \xi$ of the data on a randomly chosen pair of orthogonal unit vectors $w_\pm$. The right panel corresponds to the projections $b_\pm = B_\pm \cdot \xi$, diamonds mark the position of the cluster centers.

The conditional densities $P(\xi \mid \sigma{=}\pm 1)$ correspond to isotropic Gaussian clusters with variances v_σ centered at λB_σ. The cluster weights or prior probabilities p_σ satisfy the condition $p_+ + p_- = 1$. We assume the center vectors B_σ to be orthonormal, that is, $B_+^2 = B_-^2 = 1$ and $B_+ \cdot B_- = 0$ in the following. While the distance of the cluster centers is controlled by the model parameter $\lambda = O(1)$, the orthogonality condition fixes their position with respect to the origin. The target classification is taken to coincide with the cluster label $\sigma = \pm 1$. Note that, due to the overlap of the clusters, the task is obviously not linearly separable.

Statistical physics has been used earlier to analyse the generalization behavior of perceptron type learning prescriptions in similar models. We refer to the work of Barkai et al. (1993) and Meir (1995) for examples. The former considers learning from data in the limit of large separations $\lambda \gg v_+, v_-$. The latter compares the off-line minimization of the training error with a maximum-likelihood estimation of the cluster centers. The learning of a linearly separable rule which is defined for clustered input data has been studied in the work of Marangi et al. (1995) and Riegler et al. (1996). Here we are interested in the dynamics and generalization ability of LVQ-type learning rules for non-separable data.

In the following we denote conditional averages over $P(\xi \mid \sigma)$ by $\langle \cdots \rangle_\sigma$ whereas $\langle \cdots \rangle = \sum_{\sigma=\pm1} p_\sigma \langle \cdots \rangle_\sigma$ represents mean values with respect to the full density (1). According to Eq. (1), the components ξ_j are statistically independent quantities with variance v_σ. For an input from cluster σ we have, for instance, $\langle \xi_j \rangle_\sigma = \lambda (B_\sigma)_j$ and hence

$$\begin{aligned} \langle \xi^2 \rangle_\sigma &= \sum_{j=1}^{N} \left\langle \xi_j^2 \right\rangle_\sigma = \sum_{j=1}^{N} \left(v_\sigma + \left\langle \xi_j \right\rangle_\sigma^2 \right) = v_\sigma N + \lambda^2 \\ \Rightarrow \langle \xi^2 \rangle &= (p_+ v_+ + p_- v_-)\, N + \lambda^2. \end{aligned} \tag{2}$$

In the mathematical treatment we will exploit formally the thermodynamic limit $N \to \infty$, corresponding to very high-dimensional data and prototypes. Among other simplifying consequences this allows, for instance, to neglect the term λ^2 on the right hand side of Eq. (2).

The clusters overlap significantly and, moreover, their separation is along a single direction $(B_+ - B_-)$ in the N-dimensional input space. Figure 1 displays sample data in $N = 200$ dimensions, generated according to a density of the form (1). While the clusters are clearly visible in the plane spanned by B_+ and B_-, projections into a randomly chosen two-dimensional subspace do not display the separation at all.

2.3 On-line Algorithms

We consider so-called on-line learning schemes in which a sequence of single example data $\{\xi^\mu, \sigma^\mu\}$ is presented to the system. At each learning step, the update of prototype vectors is based only on the current example. Consequently, an explicit storage of data in the learning system is avoided and the computational effort per step is low compared to costly off-line or batch prescriptions.

In many practical situations examples are drawn from a given training set with possible multiple presentations of the same data. Here, however, we will assume that at each time step $\mu = 1, 2, \ldots$ a new, uncorrelated vector ξ^μ is generated independently according to the density (1). We will treat incremental updates of the generic form

$$w_S^\mu = w_S^{\mu-1} + \Delta w_S^\mu \text{ with } \Delta w_S^\mu = \frac{\eta}{N} f_S \left[d_+^\mu, d_-^\mu, \sigma^\mu, \ldots \right] \left(\xi^\mu - w_S^{\mu-1} \right) \tag{3}$$

where the vector w_S^μ denotes the prototype after presentation of μ examples and the constant learning rate η is rescaled with the input dimension N.

Prototypes are always moved towards or away from the current input ξ^μ. The modulation function $f_S[\ldots]$ in Eq. (3) defines the actual training algorithm and controls the sign and magnitude of the update of vectors w_S. In general, f_S will depend on the prototype label S and the class label σ^μ of the example; for convenience we denote the dependence on S as a subscript rather than as an explicit argument. The modulation function can further depend on the squared Euclidean distances of ξ^μ from the current positions of the prototypes:

$$d_S^\mu = \left(\xi^\mu - w_S^{\mu-1} \right)^2 .$$

Additional arguments of f_S, for instance the lengths or relative positions of the vectors $w_S^{\mu-1}$, could be introduced easily.

The use of a different learning rate per class or even per prototype can be advantageous in practice. This is expected, in particular, if the training data is highly unbalanced with respect to the class membership of examples. Here, we restrict the analysis to using a unique value of η, for simplicity. As we will see, the behavior of algorithms is qualitatively the same in large ranges of prior weights $p_\pm$.

In this work we study the performance of several on-line training prescriptions which can be written in the form of Eq. (3):

a) LVQ1

Kohonen's original formulation of learning vector quantization (Kohonen, 1995, 1997) extends the concept of unsupervised competitive learning in an intuitive way to a classification

task. At each learning step, the prototype with minimal distance from the current example is determined. Only this so-called *winner* is updated, hence the term *winner-takes-all* (WTA) algorithms has been coined for this type of prescription. The winner is moved towards (away from) the example input ξ^μ if prototype label and class membership of the data agree (differ), respectively.
This plausible strategy realizes a compromise between (I) the representation of data by prototypes with the same class label and (II) the identification of decision boundaries which clearly separate the different classes.
LVQ1 for two prototypes is defined by Eq. (3) with the modulation function

$$f_S[d_+^\mu, d_-^\mu, \sigma^\mu] = \Theta\left(d_{-S}^\mu - d_{+S}^\mu\right) S\sigma^\mu \quad \text{with} \quad \Theta(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{else.} \end{cases} \tag{4}$$

Here, the Heaviside function singles out the winning prototype and the factor $S\sigma^\mu = \pm 1$ determines the sign of the update. Eq. (4) expands to $f_+ = \Theta(d_-^\mu - d_+^\mu)\sigma^\mu$ for the prototype representing class $S = +1$ (or '$+$', for short). Correspondingly, it reads $f_- = -\Theta(d_+^\mu - d_-^\mu)\sigma^\mu$ for prototype label $S = -1$.

Note that LVQ1 cannot be interpreted as a stochastic gradient descent procedure in a straightforward way. This is in contrast to unsupervised vector quantization (VQ), which disregards labels completely and would reduce to the choice $f_S[\ldots] = \Theta\left(d_{-S}^\mu - d_{+S}^\mu\right)$ in our model situation (Biehl et al., 1997, 2005; Ghosh et al., 2005).

b) LVQ+/-

A popular modification of basic LVQ was also suggested by Kohonen (1997, 1990) and was termed LVQ2.1. Originally it is based on heuristics and aims at a more efficient separation of prototypes which represent different classes. Given a single example, the update concerns two prototypes: (I) the closest among all codebook vectors which belong to the same class as the data, and (II) the closest vector among the prototypes that represent a different class. The so-called *correct winner* (I) is moved towards the data whereas the *wrong winner* (II) is moved even farther away.

An important ingredient of LVQ2.1 is that the update is only performed if the example ξ^μ falls into the vicinity of the current decision boundary. Variants have been suggested, such as LVQ-LR (Seo and Obermayer, 2003), which differ in the way this selective *window* is defined.

In order to obtain a first insight into this type of algorithm, we consider the unrestricted version or, in other words, the limit of infinite window size. We will refer to this basic prescription as LVQ+/-, in the following.

In our model with two prototypes only, the identification of the respective winners is redundant and the prescription reduces to updating both vectors from each example. The modulation function in Eq. (3) merely determines the sign of Δw_S according to the class label:

$$f_S[\sigma^\mu] = S\sigma^\mu = \pm 1. \tag{5}$$

Algorithms like LVQ2.1 or LVQ-LR were suggested in order to improve the achieved generalization ability in comparison with basic LVQ1. It is well known, however, that a suitable window size has to be chosen in order to avoid divergence and stability problems. We will demonstrate for unrestricted LVQ+/- that these difficulties occur already in our simplifying model scenario. Various strategies have been suggested in the literature to cope with this problem while keeping the basic form of the algorithm. Here, we will focus on a so-called *early stopping* strategy, which aims at ending the learning process as soon as the generalization ability begins to deteriorate due to the instability.

c) LFM

Recently, Seo and Obermayer suggested a cost function for LVQ which is based on *likelihood ratios* (Seo and Obermayer, 2003). They derive the corresponding *robust soft learning vector quantization* (RSLVQ) scheme which is supposed to overcome the above mentioned stability problems of LVQ+/-. It has been demonstrated that RSLVQ has the potential to yield very good generalization behavior in practical situations (Seo and Obermayer, 2003).

The *crisp* version of RSLVQ is particularly transparent: In the limiting case of deterministic assignments an update analogous to LVQ+/- is performed only if the current configuration of prototypes misclassifies the new example. If, on the contrary, the correct winner is indeed the closest of all prototypes, the configuration is left unchanged. In analogy to perceptron training (e.g., Watkin et al., 1993; Engel and van den Broeck, 2001), we use the term *learning from mistakes* (LFM) for this prescription.

In our model scenario LFM is implemented by the choice

$$f_S[d_+^\mu, d_-^\mu, \sigma^\mu] = \Theta\left(d_{\sigma^\mu}^\mu - d_{-\sigma^\mu}^\mu\right) S\sigma^\mu. \tag{6}$$

Here, the Heaviside function restricts the update to examples which are misclassified upon presentation.

d) LFM-W

The LFM scheme (6) can also be obtained as a limiting case of Kohonen's LVQ2 algorithm, which includes a window rule for the selection of data (Kohonen et al., 1988). As shown by Bottou (1991), this prescription can be interpreted as a stochastic gradient descent with respect to a proper cost function.

As in LFM, an update of the *correct winner* w_{σ^μ} and the *wrong winner* $w_{-\sigma^\mu}$ is only performed if the current classification is wrong, that is, $d_{-\sigma^\mu}^\mu < d_{\sigma^\mu}^\mu$ for an example from class σ^μ. Here, however, the additional constraint

$$\left(\xi^\mu - w_{\sigma^\mu}\right)^2 - \left(\xi^\mu - w_{-\sigma^\mu}\right)^2 \quad < \quad c\left(\xi^\mu - w_{-\sigma^\mu}\right)^2 \tag{7}$$

has to be satisfied, where c is a small positive number (Kohonen et al., 1988; Bottou, 1991). Hence, the vector ξ^μ is only accepted for update if it is not too far away from the correct winner. In other words, ξ^μ has to be misclassified and fall into a window in the vicinity of the current decision boundary.

For the type of high-dimensional data we consider here, the window size c has to satisfy a particular scaling to be meaningful. While the term $(\xi^\mu)^2 = O(N)$ cancels on the left hand side (l.h.s.) of Eq. (7), it dominates the r.h.s. in the limit $N \to \infty$ and the condition is non-trivial only if $c = O(1/N)$. Hence, we introduce a rescaled parameter $\delta = c\,(\xi^\mu)^2 = O(1)$ and obtain $0 < (d_{\sigma^\mu}^\mu - d_{-\sigma^\mu}^\mu) < \delta$ as the conditions for non-zero updates. The first one corresponds to the misclassification of ξ^μ, the second implements the window selection in the limit $N \to \infty$.

We refer to this algorithm as LFM with window (LFM-W) in the following. It is represented by the modulation function

$$f_S[d_+^\mu, d_-^\mu, \sigma^\mu] = \left[\Theta\left(d_{\sigma^\mu}^\mu - d_{-\sigma^\mu}^\mu\right) - \Theta\left(d_{\sigma^\mu}^\mu - d_{-\sigma^\mu}^\mu - \delta\right)\right] S\sigma^\mu. \qquad (8)$$

Unrestricted LFM (6) is, of course, recovered in the limit of infinite window size $\delta \to \infty$.

2.4 Relevance of the Model

Before we proceed with the mathematical analysis, we would like to put our model into perspective and discuss its relevance.

Obviously, the learning scenario under consideration is very much simplifying and clear cut. It represents perhaps the simplest non-trivial situation in which LVQ like learning should be applicable. The presence of only one, spherically symmetric Gaussian cluster of data per class is certainly far away from practical situations encountered in typical applications. The use of only two prototype vectors is perfectly appropriate in such a scenario, and the problem of model selection is completely avoided.

Nevertheless, our model represents an important aspect of more realistic multi-class multi-prototype situations: the competition of the two prototypes which currently define the decision boundary or a portion thereof. Note that many practical algorithms reduce to the modification of only one or two prototype vectors in every single update step. Note furthermore that the model also allows to investigate unsupervised vector quantization (VQ), which is equivalent to the competition of two prototypes within the same class in our framework (see, for example, Biehl et al., 1997 and Ghosh et al., 2004).

As we will demonstrate in the following, highly non-trivial properties of the considered training algorithms will become apparent in the course of our analysis. While additional phenomena and complications must be expected in more complex training scenarios, the effects observed here will certainly persist and play an important role under more general circumstances.

Some of the training algorithms we consider are simplifying limit cases of prescriptions suggested in the literature. For example, the above defined LFM algorithm can be interpreted as the crisp limit of RSLVQ (Seo and Obermayer, 2003). Obviously, results for the former will not apply immediately to the latter. Nevertheless, we aim at a principled understanding of how certain ingredients of an algorithm influence its learning dynamics and generalization behavior. The ultimate goal is of course to exploit this understanding in the optimization of existing schemes and the development of novel, efficient ones.

3. Mathematical Analysis

We apply the successful theory of on-line learning (see, for example, Biehl and Caticha, 2003; Engel and van den Broeck, 2001; Saad, 1999, for reviews) to describe the dynamics of LVQ algorithms. The mathematical treatment of our model is based on the assumption of high-dimensional data and prototypes $\xi, w_\pm \in \mathbb{R}^N$. The thermodynamic limit $N \to \infty$ allows to apply concepts from statistical physics in the following key steps which will be detailed below:

1. The original system including many degrees of freedom is characterized in terms of only a few quantities, so-called *macroscopic order parameters*. For these, recursion relations can be derived from the learning algorithm.

2. Applying the central limit theorem enables us to perform an average over the random sequence of example data by means of Gaussian integrations.

3. Self-averaging properties of the order parameters allow to restrict the description to their mean values. Fluctuations of the stochastic dynamics can be neglected in the limit $N \to \infty$.

4. A *continuous time limit* leads to the description of the dynamics in terms of coupled, deterministic ordinary differential equations (ODE) for the above mentioned order parameters.

5. The (numerical) integration of the ODE for given modulation function and initial conditions yields the evolution of order parameters in the course of learning. From the latter one can directly obtain the learning curve, that is, the generalization ability of the LVQ classifier as a function of the number of example data.

3.1 Characteristic Quantities and Recursions

The selection of meaningful macroscopic quantities reflects, of course, the particular structure of the model. After presentation of μ examples, the system is characterized by the high-dimensional vectors w_+^μ, w_-^μ and their positions relative to the center vectors B_+, B_-. As we will demonstrate in the following, a suitable set of order parameters is

$$R_{S\sigma}^\mu = w_S^\mu \cdot B_\sigma \quad \text{and} \quad Q_{ST}^\mu = w_S^\mu \cdot w_T^\mu \quad \text{with} \quad \sigma, S, T \in \{-1, +1\}.$$

The self-overlaps Q_{++}^μ, Q_{--}^μ and the symmetric cross-overlap $Q_{+-}^\mu = Q_{-+}^\mu$ relate to the lengths and relative angle between the prototype vectors, whereas the four quantities $R_{S\sigma}^\mu$ specify their projections into the subspace spanned by $\{B_+, B_-\}$.

From the generic algorithm (3) we can directly derive recursion relations which specify the change of order parameters upon presentation of example ξ^μ from cluster σ^μ:

$$\begin{aligned}
\frac{R_{S\sigma}^\mu - R_{S\sigma}^{\mu-1}}{1/N} &= \eta f_S \left(B_\sigma \cdot \xi^\mu - R_{S\sigma}^{\mu-1} \right), \\
\frac{Q_{ST}^\mu - Q_{ST}^{\mu-1}}{1/N} &= \eta f_S \left(w_T^{\mu-1} \cdot \xi^\mu - Q_{ST}^{\mu-1} \right) + \eta f_T \left(w_S^{\mu-1} \cdot \xi^\mu - Q_{ST}^{\mu-1} \right) \\
&\quad + \eta^2 f_S f_T \, (\xi^\mu)^2 / N.
\end{aligned} \tag{9}$$

Here we use the shorthand f_S for the modulation function of prototype S, omitting the arguments.

3.2 Average over Random Examples

For a large class of modulation functions, including the ones considered here, the current input ξ^μ appears on the right hand side of Eq. (9) only through its length and the projections

$$h_S^\mu = w_S^{\mu-1} \cdot \xi^\mu \quad \text{and} \quad b_\sigma^\mu = B_\sigma \cdot \xi^\mu. \tag{10}$$

Note that also the Heaviside terms as they appear in the modulation functions, Eqs. (4,6,8), do not depend on ξ^μ explicitly, for example:

$$\Theta\left(d_-^\mu - d_+^\mu\right) = \Theta\left(+2(h_+^\mu - h_-^\mu) - Q_{++}^{\mu-1} + Q_{--}^{\mu-1}\right).$$

When performing the average over the current example ξ^μ we first exploit Eq. (2) which yields

$$\lim_{N\to\infty} \frac{1}{N} \left\langle (\xi^\mu)^2 \right\rangle = (v_+ p_+ + v_- p_-)$$

for all input vectors in the thermodynamic limit.

We assume that the new example input ξ^μ is uncorrelated with all previous data and, thus, also with the current vectors $w_S^{\mu-1}$. Therefore, the central limit theorem implies that the projections, Eq. (10), become correlated Gaussian quantities for large N. Note that this observation does not rely on the fact that the specific density (1) is a mixture of Gaussians itself. It holds whenever the components of ξ^μ are statistically independent and have the same class conditional first and second moments as in (1).

The joint Gaussian density $P(h_+^\mu, h_-^\mu, b_+^\mu, b_-^\mu)$ is fully specified by first and second moments. As shown in the appendix, for an input from cluster σ these read

$$\left\langle h_S^\mu \right\rangle_\sigma = \lambda R_{S\sigma}^{\mu-1}, \quad \left\langle b_\tau^\mu \right\rangle_\sigma = \lambda \delta_{S\tau}, \quad \left\langle h_S^\mu h_T^\mu \right\rangle_\sigma - \left\langle h_S^\mu \right\rangle_\sigma \left\langle h_T^\mu \right\rangle_\sigma = v_\sigma Q_{ST}^{\mu-1},$$

$$\left\langle h_S^\mu b_\tau^\mu \right\rangle_\sigma - \left\langle h_S^\mu \right\rangle_\sigma \left\langle b_\tau^\mu \right\rangle_\sigma = v_\sigma R_{S\tau}^{\mu-1}, \quad \left\langle b_\rho^\mu b_\tau^\mu \right\rangle_\sigma - \left\langle b_\rho^\mu \right\rangle_\sigma \left\langle b_\tau^\mu \right\rangle_\sigma = v_\sigma \delta_{\rho\tau} \tag{11}$$

where $S, T, \sigma, \rho, \tau \in \{+1, -1\}$ and $\delta_{\ldots}$ is the Kronecker-Delta. Hence, the density of $h_\pm^\mu$ and $b_\pm^\mu$ is given in terms of the model parameters $\lambda, p_\pm, v_\pm$ and the above defined set of order parameters in the previous time step.

This important result enables us to perform an average of the recursion relations, Eq. (9), over the latest example data in terms of Gaussian integrations. Moreover, the result can be expressed in closed form in $\{R_{S\sigma}^{\mu-1}, Q_{ST}^{\mu-1}\}$. In the appendix we detail the calculation and give specific results for the algorithms considered here.

3.3 Self-averaging Property

The thermodynamic limit has an additional simplifying consequence: Fluctuations of the quantities $\{R_{S\sigma}^\mu, Q_{ST}^\mu\}$ decrease with increasing N and in fact vanish for $N \to \infty$. Hence, a description in terms of their mean values is sufficient. In the statistical physics of disordered systems the term *self-averaging* is used for this property.

For a detailed mathematical discussion of self-averaging in on-line learning we refer to the work of Reents and Urbanczik (1998). Here, we have confirmed this property empirically in Monte Carlo simulations of the learning process by comparing results for various values of N, see Figure 2 (right panel) and the discussion in Section 3.6.

Neglecting fluctuations allows us to interpret the averaged recursion relations directly as recursions for the means of $\{R_{S\sigma}^\mu, Q_{ST}^\mu\}$ which coincide with their actual values in very large systems.

3.4 Continuous Learning Time

An essentially technical simplification is due to the fact that, for $N \to \infty$, we can interpret the differences on the left hand sides of Eq. (9) as derivatives with respect to the continuous learning time

$$\alpha = \mu / N.$$

Clearly, one would expect that the number of examples required for successful training should grow linearly with the number of adjustable parameters in the system. Hence, the rescaling of μ with the dimension of the feature and prototype space appears natural.

The resulting set of coupled ODE obtained from Eq. (9) is of the form

$$\begin{aligned}
\frac{dR_{S\tau}}{d\alpha} &= \eta\left(\langle b_\tau f_S\rangle - R_{S\tau}\langle f_S\rangle\right), \\
\frac{dQ_{ST}}{d\alpha} &= \eta\left(\langle h_S f_T + h_T f_S\rangle - Q_{ST}\langle f_S + f_T\rangle\right) \\
&\quad + \eta^2 \sum_{\sigma=\pm1} v_\sigma p_\sigma \langle f_S f_T\rangle_\sigma .
\end{aligned} \tag{12}$$

The required averages and specific equations for the particular modulation functions considered here are given in the Appendix, Sections A.3 and A.4. Note that the ODE for Q_{ST} contain terms proportional to η and η^2 while $dR_{S\tau}/d\alpha$ is linear in the learning rate.

3.5 The Learning Curve

After working out the system of ODE for a specific modulation function, it can be integrated, at least numerically. For given initial conditions $\{R_{S\sigma}(0), Q_{ST}(0)\}$, learning rate η, and model parameters $\{p_\pm, v_\pm, \lambda\}$ one obtains the typical evolution $\{R_{S\sigma}(\alpha), Q_{ST}(\alpha)\}$. This is the basis for analysing the performance and convergence properties of various algorithms in the following.

Most frequently, we will consider prototypes that are initialized as independent random vectors of squared length $\widehat{Q}$ with no prior knowledge about the cluster positions. In terms of order parameters this implies in our model

$$Q_{++}(0) = Q_{--}(0) = \widehat{Q},\;\; Q_{+-}(0) = 0 \;\;\text{and}\;\; R_{S\sigma}(0) = 0 \;\;\text{for all}\;\; S, \sigma = \pm 1. \tag{13}$$

Obviously, the precise initial positions of prototypes with respect to the cluster geometry can play a crucial role for the learning dynamics. In the next section we will demonstrate that the outcome of, for instance, LFM and LVQ+/- with early stopping can depend strongly on the initial positions of prototypes. On the contrary, asymptotic properties of, for example, LVQ1 training will prove independent of initialization in the limit of infinitely long training sequences within our model situation.

After training, the success of learning can be quantified in terms of the generalization error, that is, the probability for misclassifying novel, random data which did not appear in the training sequence. Here we can consider the two contributions for misclassifying data from cluster $\sigma = 1$ or $\sigma = -1$ separately:

$$\varepsilon = p_+\,\varepsilon_+ + p_-\,\varepsilon_- \quad \text{with} \quad \varepsilon_\sigma = \langle \Theta\left(d_{+\sigma} - d_{-\sigma}\right)\rangle_\sigma . \tag{14}$$

Exploiting the central limit theorem in the same fashion as above, one can work out the generalization error as an explicit function of the order parameters. As detailed in the appendix, Section A.6, one obtains for the above contributions $\varepsilon_\pm$:

$$\varepsilon_\sigma = \Phi\left(\frac{Q_{\sigma\sigma} - Q_{-\sigma-\sigma} - 2\lambda(R_{\sigma\sigma} - R_{-\sigma\sigma})}{2\sqrt{v_\sigma}\sqrt{Q_{++} - 2Q_{+-} + Q_{--}}}\right) \quad \text{where} \quad \Phi(z) = \int_{-\infty}^{z} dx\, \frac{e^{-x^2/2}}{\sqrt{2\pi}}. \tag{15}$$

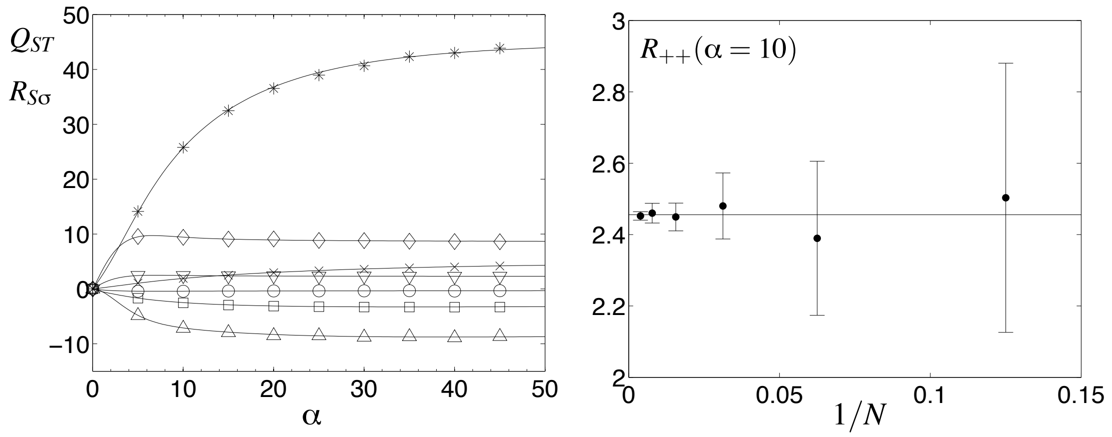


Figure 2: LVQ1 with $\lambda = 2, v_+ = 4, v_- = 9, p_+ = 0.8$, and learning rate $\eta = 1.0$.
Left panel: Characteristic overlaps vs. $\alpha = \mu/N$. Solid lines display the result of integrating the system of ODE for initial conditions as in Eq. (13) with $\widehat{Q} = 10^{-4}$. Symbols represent Monte Carlo simulations for $N = 100$, on average over 100 independent runs. Standard error bars would be smaller than the symbol size. Curves and symbols correspond to: $Q_{--}\,(*), Q_{++}\,(\Diamond), R_{--}\,(\times), R_{++}\,(\bigtriangledown), R_{+-}\,(\bigcirc), R_{-+}\,(\Box)$, and $Q_{+-}\,(\triangle)$.
Right panel: Example for the self-averaging behavior of order parameters as observed in Monte Carlo simulations for $N = 8, 16, 32, 64, 128$, and 256. Dots mark the observed average value of $R_{++}(\alpha = 10)$ vs. $1/N$, bars represent the corresponding variance. Here, the latter vanishes approximately like $1/N$, while the mean values display no systematic dependence on N for large enough systems. The horizontal line marks the theoretical prediction for $N \to \infty$.

By inserting $\{R_{S\sigma}(\alpha), Q_{ST}(\alpha)\}$ we obtain the learning curve $\varepsilon_g(\alpha)$, that is, the typical generalization error after on-line training with αN random examples. Here, we once more exploit the fact that the order parameters and, thus, also ε_g are self-averaging, non-fluctuating quantities in the thermodynamic limit $N \to \infty$.

A classification scheme based on two prototypes is restricted to linear decision boundaries. We will therefore compare the performance of LVQ algorithms with the best linear decision (bld) boundary for given parameters $p_\pm, v_\pm$, and λ. For symmetry reasons it is given by a plane orthogonal to $(B_+ - B_-)$. It is straightforward to obtain the corresponding generalization error ε_g^{bld} from Eqs. (14,15) by appropriate parameterization of the plane and minimization of ε_g (Ghosh et al., 2004).

Note that the Bayes optimal classification of data from density (1) is, in general, given by a non-linear decision boundary which contains the vectors ξ with $p_+ P(\xi|+1) = p_- P(\xi|-1)$ (Duda et al., 2000). Only for $v_+ = v_-$ it becomes a plane and LVQ with two prototypes could potentially implement Bayes optimal generalization.

3.6 Restrictions of the Analysis

While we have already discussed the relevance of our simplifying model scenario in Subsection 2.4, we want to summarize here some restrictions of the mathematical analysis.

Perhaps, the consideration of the *thermodynamic limit* $N \to \infty$ of infinite-dimensional feature vectors appears to be the most severe limitation in the mathematical treatment. Together with the assumption of statistically independent features ξ_j^μ, it facilitates the above explained steps of the analysis, the evaluation of averages over random data being the most important one.

We find that our results describe very well the (mean) behavior of systems with a fairly small number of input dimensions, yielding excellent agreement for $N = 100$ already. Fig. 2 (right panel) shows a comparison of different system sizes and illustration of the above mentioned *self-averaging* property. In other learning problems, the behavior of low-dimensional systems may differ significantly from results obtained in the limit $N \to \infty$, as fluctuations which were neglected here become more important. As an example, we mention the symmetry breaking specialization of prototypes in unsupervised VQ (Biehl et al., 1997). Here, however, the self-averaging behavior of order parameters is reflected by the fact that their variances vanish rapidly with N, approximately like $1/N$, see Figure 2 (right panel). At the same time, no systematic dependence of the means on the system size is observed. Hence, our treatment yields an excellent approximation for systems of fairly small dimension N, already: Deviations of observed and predicted values of characteristic overlaps are expected to be on the order $1/\sqrt{N}$. For analytic results concerning such *finite size corrections* in on-line training see, for instance, Saad (1999) and references therein.

Performing averages over the randomness in the data yields *typical properties* of the system in the considered model situations. This method is different in spirit and thus complements other successful approaches in computational learning theory, where one aims at rigorous bounds on the generalization error without making explicit assumptions about the learning scenario (for examples in the context of LVQ, see Crammer et al., 2003; Hammer et al., 2005a). Such bounds are not necessarily *tight* and can be quite far from the actual behavior observed in practical situations. On the contrary, results obtained in the flavor of statistical physics analysis lack the mathematical rigor of strict bounds and may be sensitive to details of the model assumptions, for example, the statistical properties of the data. As an attractive feature, the approach provides information about the system which goes beyond its generalization ability, such as the learning dynamics and the location of prototypes.

For more detailed discussions of strengths and limitations of our approach we refer to the reviews of, for example, Biehl and Caticha (2003), Watkin et al. (1993), and Engel and van den Broeck (2001).

3.7 Relation to Statistical Physics

The relation to statistical physics is not crucial for what follows and may be considered a merely technical point. Nevertheless, for the interested reader, we would like to make a few remarks in this context.

The analogy is more important in the theory of batch or off-line learning. There, the cost function of training is interpreted as an *energy* and the analysis proceeds along the lines of equilibrium statistical mechanics. For an introduction to these concepts we refer to, for example, Biehl and Caticha (2003), Watkin et al. (1993), and Engel and van den Broeck (2001). Several ideas from this type of analysis do carry over to the investigation of on-line learning which addresses the non-equilibrium dynamics of learning.

In many physical systems it is impossible, and indeed useless, to keep track of all microscopic degrees of freedom. As an example, consider the positions and velocities of, say, N particles in a gas. Similarly, a magnetic material will contain a number N of atoms each of which contributes one elementary magnetic moment. As N gets very large, say, on the order 10^{23} in condensed matter physics, microscopic details become less important and such systems are successfully described by a fairly small number of macroscopic quantities or order parameters. In the above examples, it may be sufficient to know volume and pressure of the gas or the total magnetization emerging from the collaborative behavior of atoms. Mathematically speaking, so-called phase space integrals over all degrees of freedom can be performed by means of a saddle point approximation for $N \to \infty$, reducing the description to a low-dimensional one. Similarly, the high-dimensional phase space trajectories of dynamical systems can be represented by the temporal evolution of a few macroscopic order parameters. The same idea applies here as we describe the learning dynamics of $2N$ prototype components in terms of a few characteristic overlaps and disregard their microscopic details.

An important branch of statistical physics deals with so-called *disordered systems*, where the interacting degrees of freedom are embedded in a random environment. In the above example this could mean that magnetic atoms are randomly placed within an otherwise non-magnetic material, for instance. The correct analytic treatment of such systems requires sophisticated analytical tools such as the famous *replica trick*, see Watkin et al. (1993) and Engel and van den Broeck (2001) and references therein.

These tools have been developed in the context of disordered magnetic materials, indeed, and have been put forward within the statistical physics of learning. In learning theory, the *disorder* emerges from the random generation of training data. In off-line learning, the cost-function or *energy* is defined for one specific set of data only and the generic, typical behavior is determined by performing the *disorder average* over all possible training sets. This requires the above mentioned replica method or approximation techniques which involve subtle mathematical difficulties, see Watkin et al. (1993) and Engel and van den Broeck (2001).

The situation is slightly more favorable in the framework which we resort to here: In on-line learning at each time step of the process, a novel random example is presented. As a consequence the *disorder average* can be performed step by step in the more elementary fashion outlined above (Biehl and Caticha, 2003; Engel and van den Broeck, 2001).

4. Results

In the following we present our results obtained along the lines of the treatment outlined in Sec. 3. We will compare the typical learning curves of LVQ+/-, an idealized early stopping procedure, LFM, and LFM-W with those of LVQ1. The latter, original formulation of basic LVQ serves as a reference each modification has to compete with.

We will put emphasis on the asymptotic behavior in the limit $\alpha \to \infty$, that is, the generalization error achieved from an arbitrarily large number of examples and its dependence on the model parameters. This asymptotic behavior is of particular relevance for comparing different training algorithms. It can be studied by analysing the stable fixed point configuration of the system of ODE. Note that properties thereof will not depend on initialization or the position of cluster cen-

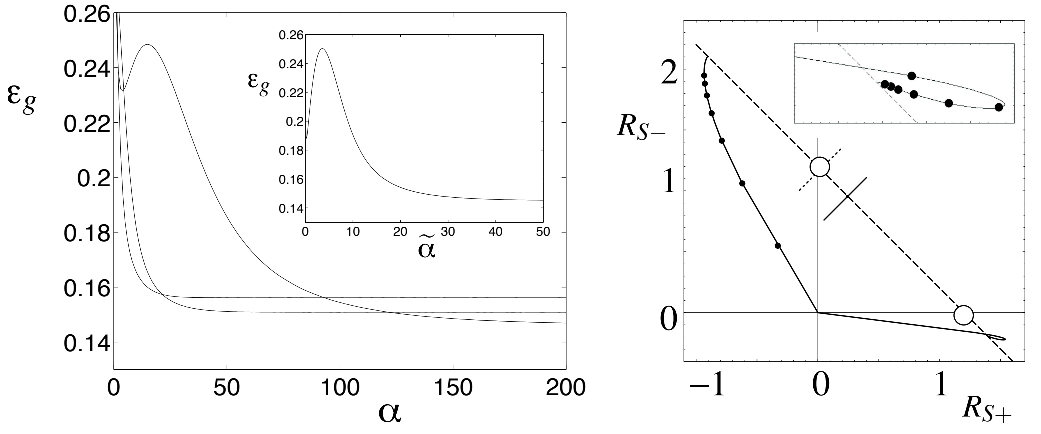


Figure 3: LVQ1 for $\lambda = 1.2, v_+ = v_- = 1$, and $p_+ = 0.8$, initialization as in Figure 2.
Left panel: Learning curves $\varepsilon_g(\alpha)$ for three different learning rates $\eta = 0.2, 1.0, 2.0$ (from bottom to top at $\alpha = 200$). Large η are favorable initially, whereas smaller η yield better asymptotic generalization for $\alpha \to \infty$. The inset shows the limiting behavior for $\eta \to 0$ and $\alpha \to \infty$, that is, ε_g as a function of $\widetilde{\alpha} = \eta\alpha$.
Right panel: Trajectories of prototypes in the limit $\eta \to 0, \alpha \to \infty$. Solid lines correspond to the projections of prototypes into the plane spanned by λB_+ and λB_- (marked by open circles). The dots correspond to the pairs of values $\{R_{S+}, R_{S-}\}$ observed at $\widetilde{\alpha} = \eta\alpha = 2, 4, 6, 8, 10, 12, 14$ in Monte Carlo simulations with $\eta = 0.01$ and $N = 200$, averaged over 100 runs. Note that, because $p_+ > p_-$, w_+ approaches its final position much faster and in fact *overshoots*. The inset displays a close-up of the region around its stationary location. The short solid line marks the asymptotic decision boundary as parameterized by the prototypes, the short dashed line marks the best linear decision boundary. The latter is very close to λB_- for the pronounced dominance of the $\sigma = +1$ cluster with $p_+ = 0.8$.

ters relative to the origin. Asymptotic properties are controlled only by the distance of the centers $|\lambda B_+ - \lambda B_-| = \sqrt{2}\lambda$, the model parameters $v_\pm, p_\pm$ and the learning rate η.

4.1 LVQ1

Figure 2 (left panel) displays the evolution of order parameters in the course of learning for an example choice of the model parameters. Monte Carlo simulations for $N = 100$ already agree very well with the $(N \to \infty)$ theoretical prediction based on integrating the corresponding ODE, Eq. (34) . We consider initialization of prototypes close to the origin, that is, relatively far from the region of high density in our data model. Note that in a WTA algorithm the initial prototypes must not coincide exactly, hence we choose random $w_\pm(0)$ with, for example, squared length $Q_{++} = Q_{--} = \widehat{Q} = 10^{-4}$ in Eq. (13).

The self-averaging property, see Section 3.3, is illustrated in Fig. 2 (right panel). In Monte Carlo simulations one observes that averages of the order parameters over independent runs approach the theoretical prediction as $N \to \infty$. At the same time, the corresponding variances vanish like $1/N$ with increasing dimension.

The typical learning curve $\varepsilon_g(\alpha)$ is in the center of our interest. Figure 3 (left panel) displays the behavior of LVQ1 for different learning rates in an example situation. The pronounced non-monotonic behavior for smaller η clearly indicates that the prescription is suboptimal: For a range of α, additional information leads to a loss of generalization ability. This effect is particularly pronounced for highly unbalanced data with, say, $p_+ \gg p_-$. The results suggest a schedule which employs large values of η initially and then decreases the learning rate with increasing α. This aspect will be investigated in greater detail in a forthcoming project, here we consider only training with constant η.

For LVQ1 we find that the stationary, asymptotic generalization error $\varepsilon_g^{stat} = \varepsilon_g(\alpha \to \infty)$ decreases with $\eta \to 0$ like

$$\varepsilon_g^{stat}(\eta) = \varepsilon_g^{lvq1} + O(\eta) \quad \text{for small} \ \ \eta.$$

Here, ε_g^{lvq1} denotes the best value achievable with LVQ1 for a given set of model parameters. This is analogous to the behavior of stochastic gradient descent procedures like VQ, where the associated cost function is minimized in the simultaneous limits of small learning rates $\eta \to 0$ and $\alpha \to \infty$, such that $\widetilde{\alpha} = \eta\alpha \to \infty$. In absence of a cost function we can still consider this limit. Terms proportional to η^2 can be neglected in the ODE, and the evolution in rescaled learning time $\widetilde{\alpha}$ becomes η-independent. The inset of Fig. 3 (left panel) shows the limiting learning curve $\varepsilon_g(\widetilde{\alpha})$. It displays a strong non-monotonic behavior for small $\widetilde{\alpha}$.

The right panel of Fig. 3 displays the trajectories of prototypes projected into the plane spanned by B_+ and B_-. Note that, as could be expected from symmetry arguments, the $(\alpha \to \infty)$-asymptotic projections of prototypes into the $B_\pm$-plane are along the axis connecting the cluster centers. Moreover, in the limit $\eta \to 0$, their stationary position lies precisely in the plane and fluctuations orthogonal to $B_\pm$ vanish. This is signaled by the fact that the order parameters for $\widetilde{\alpha} \to \infty$ satisfy $Q_{SS} = R_{S+}^2 + R_{S-}^2$, and $Q_{+-} = R_{++}R_{-+} + R_{+-}R_{--}$ which implies

$$w_S(\widetilde{\alpha} \to \infty) = R_{S+}B_+ + R_{S-}B_- \ \text{ for } S = \pm 1. \tag{16}$$

Here we can conclude that the actual prototype vectors approach the above unique configuration, asymptotically. Note that, in general, stationarity of the order parameters does not imply necessarily that $w_\pm$ converge to points in N-dimensional space. For LVQ1 with $\eta > 0$ fluctuations in the space orthogonal to $\{B_+, B_-\}$ persist even for constant $\{R_{S\sigma}, Q_{ST}\}$.

Figure 3 (right) reveals further information about the learning process. When learning from unbalanced data, for example, $p_+ > p_-$ as in the example, the prototype representing the stronger cluster will be updated more frequently and in fact *overshoots*, resulting in the non-monotonic behavior of ε_g. The use of a different learning rate per class could correct this overshooting behavior and recover the transient behavior for equal priors, qualitatively.

The asymptotic ε_g^{lvq1} as achieved by LVQ1 is typically quite close to the potential optimum ε_g^{bld}. Figure 4 displays the asymptotic generalization error as a function of the prior p_+ in two different settings of the model. In the left panel $v_+ = v_-$ whereas the right panel shows an example with different cluster variances. In the completely symmetric situation with equal variances and balanced priors, $p_+ = p_-$, the LVQ1 result coincides with the best linear decision boundary which is through $\lambda(B_+ + B_-)/2$ for this setting. Whenever the cluster-variances are different, the symmetry about $p_+ = 1/2$ is lost but $|\varepsilon_g^{lvq1} - \varepsilon_g^{bld}| = 0$ for one particular (v_+, v_-)-dependent value of $p_+ \in]0,1[$.

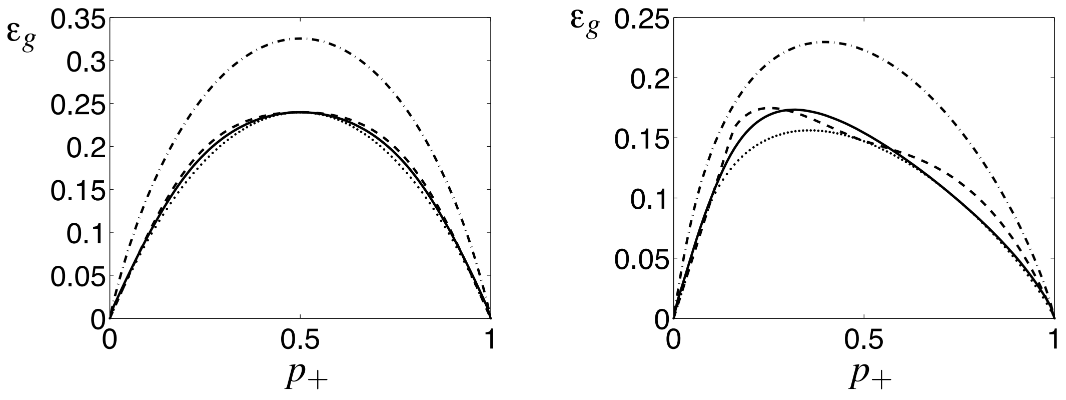


Figure 4: Achievable generalization error in the model with $\lambda = 1$ as a function of p_+. In both panels, the lowest, dotted curve represents ε_g^{bld}, that is, the best linear classifier. Solid lines mark the asymptotic ε_g^{lvq1} of LVQ1, the dashed lines corresponds to ε_g^{stop} as obtained from an idealized early stopping scheme for LVQ+/- with prototypes initialized in the origin. The far from optimal $(\alpha \to \infty)$-asymptotic ε_g^{lfm} result for LFM is marked by chain lines.
Left panel: The case of equal variances, $v_+ = v_- = 1$, ε_g^{lvq1} and ε_g^{stop} coincide for $p_+ = 1/2$ where both are optimal.
Right panel: An example for unequal variances, $v_+ = 0.25, v_- = 0.81$. The result of LVQ+/- with idealized early stopping is still optimal for equal priors, while $\varepsilon_g^{lvq1} = \varepsilon_g^{bld}$ in $p_+ \approx 0.74$ in this setting.

4.2 LVQ+/-

Here we report results concerning the divergent behavior displayed by the LVQ+/- prescription in its basic form. We furthermore show how an appropriate early stopping procedure could overcome this difficulty.

4.2.1 DIVERGENT BEHAVIOR

The structure of the ODE for LVQ+/-, $f_S = S\sigma^\mu$, is particularly simple, see Appendix A.5. Analytic integration yields Eq. (37) for settings with $p_+ \neq p_-$. In this generic case, the evolution of $\{R_{S\sigma}, Q_{ST}\}$ displays a strong divergent behavior: All quantities associated with the prototype representing the weaker cluster display an exponential increase with α.

Figure 5 (left panel) shows an example for $\lambda = 1, v_+ = v_- = 1$, learning rate $\eta = 0.5$, and unbalanced priors $p_+ = 0.8, p_- = 0.2$. Here, order parameters which involve w_-, that is, $R_{--}, R_{-+}, Q_{--}, Q_{+-}$, diverge rapidly. On the contrary, quantities which relate only to w_+, that is, R_{++}, R_{+-}, Q_{++}, remain finite and approach well-defined asymptotic values for $\alpha \to \infty$.

This behavior is due to the fact that the *wrong winner* is always moved away from the current data in LVQ+/- training. Clearly, this feature renders the learning dynamics unstable as soon as $p_+ \neq p_-$. Even in our simple model problem, repulsion will always dominate for one of the prototypes and, like w_- in our example, it will be moved arbitrarily far away from the cluster centers as $\alpha \to \infty$.

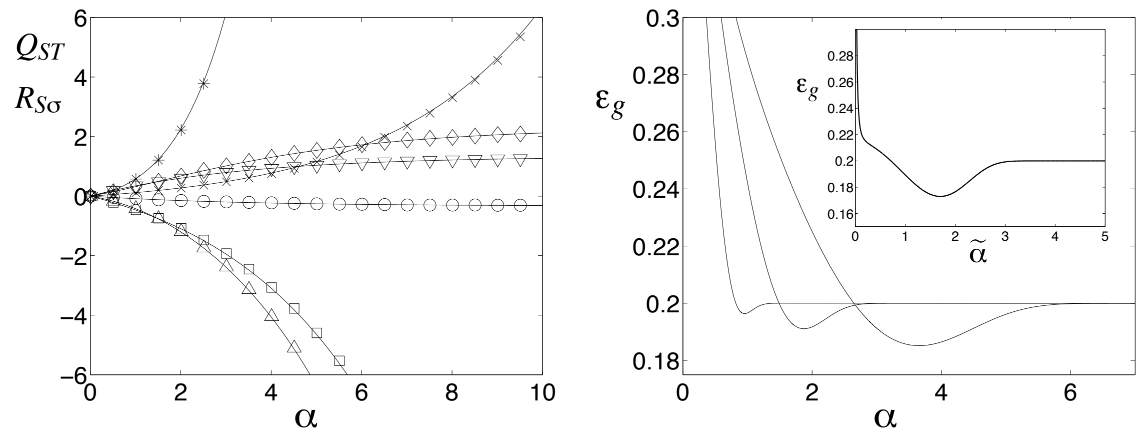


Figure 5: LVQ+/- with $\lambda = 1, v_+ = v_- = 1$, and $p_+ = 0.8$.
Left panel: Characteristic overlaps vs. α for learning rate $\eta = 0.5$. Solid lines correspond to the analytical solution Eq. (37) of the ODE for initialization $w_+(0) = w_-(0) = 0$. Symbols represent Monte Carlo results for $N = 100$ on average over 100 runs. Standard error bars would be smaller than the symbol size. Curves and symbols correspond to $Q_{--}\,(*), Q_{++}\,(\Diamond), R_{--}\,(\times), R_{++}\,(\bigtriangledown), R_{+-}\,(\bigcirc), R_{-+}\,(\Box)$, and $Q_{+-}\,(\bigtriangleup)$.
Right panel: Learning curves $\varepsilon_g(\alpha)$ for three different rates $\eta = 2.0, 1.0$, and 0.5 (from left to right). The generalization error displays a pronounced minimum at intermediate α and approaches the trivial value $\varepsilon_g = \min\{p_+, p_-\}$ asymptotically for $\alpha \to \infty$. The inset shows the asymptotic behavior $\varepsilon_g(\widetilde{\alpha})$ in the simultaneous limit $\eta \to 0, \alpha \to \infty$ with rescaled $\widetilde{\alpha} = \eta\alpha$.

The divergent behavior is also apparent in Figure 6, which displays example trajectories of $w_\pm$ in the limit $\eta \to 0$, projected into the $B_\pm$-plane. The projections of prototypes follow straight lines, approximately. Prototype w_+, representing the stronger cluster in this case, approaches a stationary position at the symmetry axis of the density (1). For w_- the less frequent positive updates from the weaker cluster $\sigma = -1$ cannot counterbalance the repulsion.

The resulting classification scheme for $\alpha \to \infty$ is trivial: All data will be assigned to the class of the stronger cluster. Hence, the asymptotic generalization error of unmodified LVQ+/- is given by $\min\{p_+, p_-\}$ in our model, independent of the learning rate η. This can also be seen in the learning curves displayed in Figure 5 (right panel).

Note that the behavior is qualitatively different for the singular case of balanced priors $p_+ = p_- = 1/2$, see Eq. (38) in the appendix for the corresponding analytic solution of the ODE. First, the increase of order parameters with α, which is exponential for $p_+ \neq p_-$, becomes linear ($R_{S\sigma}$) or quadratic (Q_{ST}) in α for equal priors. Second, both prototypes move to infinity, that is, away from the data, as $\alpha \to \infty$. In a visualization of the learning dynamics in the spirit of Figure 6, the trajectories become parallel to the symmetry axis as $p_+ = p_-$. It is interesting to note that, in spite of this divergence, the corresponding decision boundary is optimal in the singular case of equal priors.

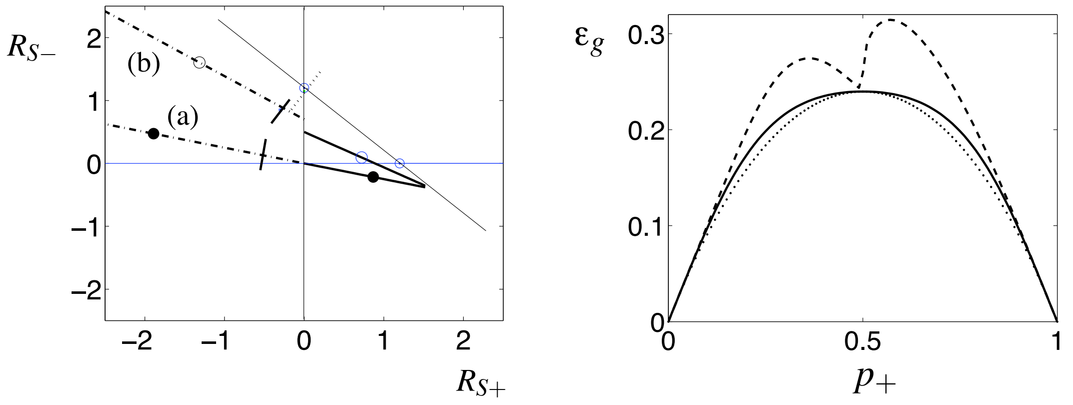


Figure 6: LVQ+/- and early stopping for different initializations in the limit $\eta \to 0, \alpha \to \infty$.
Left panel: Projected trajectories for model parameters $\lambda = 1.2, p_+ = 0.8, v_+ = v_- = 1$. Cluster centers correspond to open circles, the short dotted line marks the best linear decision boundary. Solid (chain) lines show the trace of prototype w_+ (w_-), respectively. Filled symbols display the position after *early stopping* in the minimum of $\varepsilon_g(\widetilde{\alpha})$, short solid lines mark the projection of the respective decision boundaries. Squares correspond to the positions obtained from $w_\pm(0) = 0$ (a), full circles mark the results for initialization (b) with an offset from the origin and the (B_+, B_-)-plane: $R_{++} = R_{-+} = 0, R_{+-} = 0.5, R_{--} = 0.7, Q_{+-} = 0, Q_{++} = 1.8, Q_{--} = 2.9$. In both cases, w_+ approaches the same stationary position on the symmetry axis, while w_- displays divergent behavior as $\widetilde{\alpha} = \eta\alpha$ increases. Note that the decision boundary obtained by early stopping in case (b) appears to be close to optimal in the projection. However, it is tilted strongly in the remaining dimensions as indicated by the violation of condition (16). Consequently, ε_g^{stop} is higher for (b) than it is for initialization (a).
Right panel: Generalization error ε_g^{stop} of idealized early stopping as a function of p_+ for $\lambda = 1$ and equal variances $v_+ = v_- = 1$. The dotted line corresponds to the best linear decision boundary. The solid line marks the outcome of LVQ+/- with early stopping when prototypes are initialized in the origin, case (a) in the left panel, which is also displayed in Fig. 4. The dashed line represents the far from optimal ε_g^{stop} for an example initialization with an offset from the origin, case (b) in the left panel.

4.2.2 EARLY STOPPING

Several heuristic strategies have been suggested in the literature to cure the divergent behavior while keeping the essential ingredients of LVQ+/-:

One possible measure that comes to mind immediately is to let the update of the *wrong winner* depend explicitly on its distance d^μ from the data. The divergence should be much weaker or even seize, provided the magnitude of the update decreases fast enough with d^μ or if it is cut off at a maximum distance d_{max}.

Another popular strategy is to update only from data that falls into a *window* close to the current decision boundary, that is, close to the midplane between the prototypes (Kohonen, 1997).

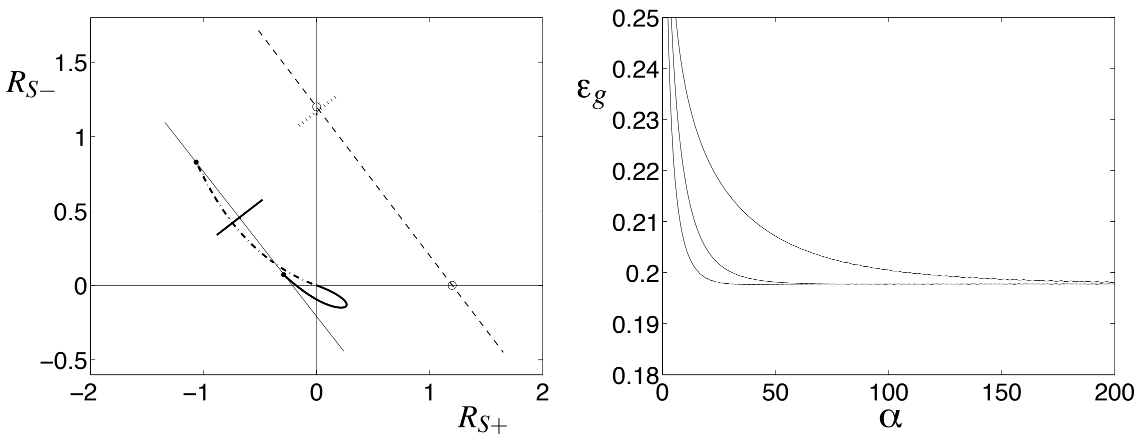


Figure 7: Learning from mistakes (LFM)
Left panel: Projected prototype trajectories for LFM with $\eta = 1$ and all other parameters as in Fig. 6 (left panel). The initial behavior for small α is similar to that of LVQ+/-. Asymptotically, both prototypes approach stationary positions (solid dots) which are not on the symmetry axis of the distribution (dashed line). Note that only the projections of $(w_+ - w_-)$ become parallel to $\lambda(B_+ - B_-)$ as $\alpha \to \infty$. While the actual $w_\pm(\alpha \to \infty)$ depend on η as discussed in the text, the asymptotic decision boundary (short solid line) does not.
Right panel: $\varepsilon_g(\alpha)$ for LFM with model parameters $\lambda = 3, p_+ = 0.8, v_+ = 4, v_- = 9$ and learning rates $\eta = 2.0, 1.0, 0.5$ (from left to right). All learning curves approach the same η-independent asymptotic value ε_g^{lfm}. Note the offset on the ε_g- axis.

Provided the decision boundary becomes similar to the Bayes optimal one in the course of learning, examples ξ from this region would satisfy $p_+P(\xi|+1) \approx p_-P(\xi|-1)$, then. In effect, the system would be trained from balanced data, which significantly slows down the divergence, see the above considerations for LVQ+/- with $p_+ = p_-$.

In principle, these strategies can also be studied within our framework by considering appropriate modifications of the modulation function, Eq. (5). Preliminary results indicate, however, that both ideas improve convergence and generalization ability of LVQ+/- only to a very limited extent. In addition, one or several parameters, for example, the cut-off distance d_{max} or the *window size*, have to be carefully tuned to achieve the desired effect.

Moreover, both variants are generically outperformed by a conceptually simpler *early stopping* strategy. The idea is to stop the learning process as soon as the divergent behavior starts to increase the generalization error. This does not require the fine-tuning of additional parameters. However, in any practical situation, one would have to monitor ε_g in a test set of data which is not used for training.

Generic learning curves of LVQ+/-, see Figure 5, display a pronounced minimum before they approach the trivial asymptotic value $\varepsilon_g = \min\{p_+, p_-\}$. We are interested in the best generalization error ε_g^{stop} that could be achieved, in principle, by an idealized early stopping method. In contrast to $(\alpha \to \infty)$-asymptotic properties, the dynamics for intermediate α will depend on initialization, as discussed below. This concerns also the depth of the minimum, which can occur at rather small α. For the cluster geometry considered here and initialization as given in Eq. (13), we find that the lowest values of ε_g are indeed achieved for $\widehat{Q} = 0$. In this setting the effect of the first examples is a very fast alignment of $(w_+ - w_-)$ with the symmetry axis of the model density (1). Hence, we first consider prototypes which are initialized precisely in the origin $w_+(0) = w_-(0) = 0$ and without offset from the plane spanned by B_+ and B_-. We observe furthermore that the respective value of ε_g in the minimum decreases with decreasing η. For the comparison with other algorithms we will therefore resort to the simultaneous limit $\eta \to 0, \alpha \to \infty$ as for LVQ1.

In our model, it is straightforward to obtain precisely the minimum of the limiting $\varepsilon_g(\widetilde{\alpha})$ from the analytic solution, Eq. (37). The result ε_g^{stop} is displayed as a function of p_+ in Figure 4 for two different choices of v_+ and v_-. In the case of equal variances $v_+ = v_-$ we find that LVQ+/- with early stopping is relatively close to the best possible ε_g^{bld}. However, it is outperformed by the asymptotic result of LVQ1 for all p_+. Note that both algorithms yield the optimal classification scheme in the singular case $p_+ = p_- = 1/2$ and $v_+ = v_-$.

For data with $v_+ \neq v_-$, we find that $\varepsilon_g^{stop} > \varepsilon_g^{lvq1}$ for small and large values of p_+, see Figure 4 (right panel) for an example. In the case of balanced priors, the learning curve of LVQ+/- does not display a minimum but approaches the optimal value ε_g^{bld} quite rapidly. For cluster weights in an interval around $p_+ = 1/2$, the result of the early stopping procedure is superior to that of LVQ1.

It is important to note that our analysis refers to an idealized early stopping procedure based on perfect knowledge of the current ε_g. On the contrary, the basic LVQ1 gives close to optimal performance independent of initial conditions, without adjustment of parameters and without explicit estimation of the generalization error.

Our results show that LVQ+/- with favorable initialization outperforms other algorithms with respect to behavior for small α, that is, with respect to learning from small training sets. Note, for instance, that the minima in the learning curve $\varepsilon_g(\alpha)$ as displayed in Fig. 5 can occur at very small α, already. While LVQ1, for example, achieves better $\alpha \to \infty$ asymptotic generalization, a comparison of the learning curves shows that it is often inferior for small and intermediate values of α. A systematic comparison is difficult, however, since the effect will depend strongly on the considered initialization. Nevertheless, the use of LVQ+/- type updates in the early stages of training or for limited availability of example data appears promising.

The crucial influence of the initialization is illustrated in Fig. 6. As an example, we consider the initialization of prototypes with an offset from the origin and, more importantly, from the plane spanned by the cluster centers. Again, learning curves display a minimum in the generalization error. However, the obtained best value of ε_g^{stop} is far from optimal, as displayed in Fig. 6 (right panel). Asymptotic properties of the considered algorithms are independent of initial settings and, hence, represent generic behavior in the frame of our model situation. On the contrary, the above discussed results for *early stopping* are highly sensitive with respect to initialization and demonstrate, at best, the potential usefulness of the strategy.

4.3 Learning from Mistakes

In the following, results are presented for the LFM training algorithm with and without a selective window.

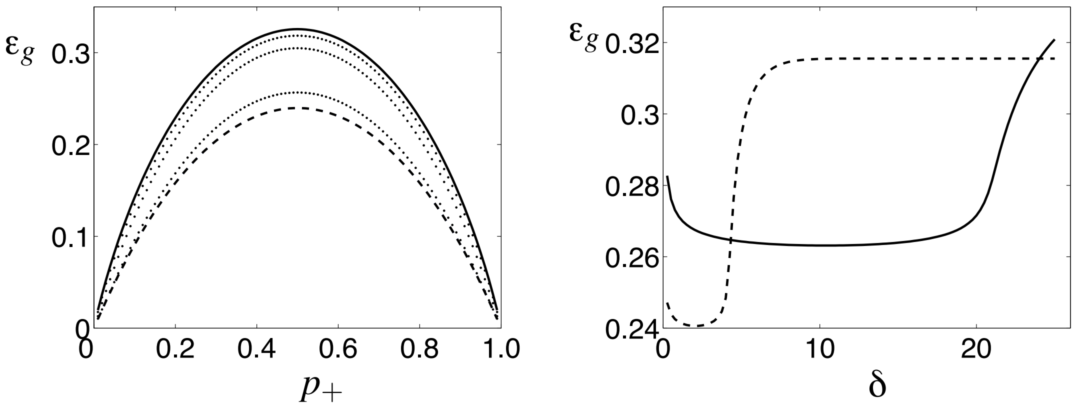


Figure 8: Learning from mistakes with window (LFM-W)
Left panel: Learning from equal variance clusters with $v_+ = v_= - 1, \lambda = 1$, and learning rate $\eta = 1$. Achieved $(\alpha \to \infty)$ asymptotic generalization error ε_g^{lfm-w} as a function of p_+ for different rescaled window sizes; from top to bottom: $\delta \to \infty$ (solid line, LFM), $\delta = 6$, $\delta = 5$, and $\delta = 0.25$ (dotted lines). The lowest, dashed line marks the best possible ε_g^{bld}.
Right panel: Asymptotic generalization error ε_g^{lfm-w} as a function of the window size δ for prior weight $p_+ = 0.6$ and learning rate $\eta = 1$. The dashed line corresponds to data with $v_+ = v_- = 1$ and $\lambda = 1$, the solid line refers to $v_+ = v_- = 5$ and $\lambda = 2$. Already in symmetric settings, the location of the optimal window size depends strongly on the properties of the data. Note the offset on the ε_g-axis.

4.3.1 LFM

The basic idea of the LFM procedure, cf. Section 2.3 (c), is reminiscent of many prescriptions for perceptron training (e.g., Watkin et al., 1993; Engel and van den Broeck, 2001; Biehl and Caticha, 2003). An LVQ+/- type of update is performed only if the current configuration would misclassify the example. Numerical integration of the corresponding ODE, Eq. (36), shows that LFM does not display the divergent behavior of LVQ+/-.

In contrast to LVQ1 and LVQ+/-, the typical learning curves of LFM display a monotonic decrease of $\varepsilon_g(\alpha)$, see the right panel of Figure 7 for examples with three different learning rates. An important feature is that the $(\alpha \to \infty)$-asymptotic generalization error of LFM learning does not depend on η.

Figure 7 (left panel) shows example trajectories of the prototypes in the $B_\pm$-plane. The behavior for small α is similar to that of LVQ+/-: Prototypes move away from the origin in opposite directions, initially. However, the attraction to (or repulsion from) the cluster centers becomes less important in the course of learning. Emphasis is on correct classification of the data, the aspect of cluster representation in the sense of VQ is essentially absent.

Eventually, the projections of the prototypes assume positions along a line which is parallel to the symmetry axis $\lambda(B_+ - B_-)$. However, the violation of condition (16) indicates that w_+ and w_- themselves do not lie in the two-dimensional subspace spanned by the cluster centers. Here, the stationarity of order parameters and generalization error does not imply convergence of the prototype vectors themselves. Even in the limit $\alpha \to \infty$, they will fluctuate within the subspace of $\mathbb{R}^N \times \mathbb{R}^N$ that is compatible with the asymptotic values of $\{R_{S\sigma}, Q_{ST}\}$.

Clearly, the precise trajectory and the final outcome of LFM depends on the initialization of prototypes. It will determine, for instance, the relative position of w_+ and w_- parallel to the connection of cluster centers. The actual asymptotic configuration of order parameters and prototypes depends furthermore on the learning rate η. However, all these configurations share the same generalization error. The learning rate merely controls the magnitude of the fluctuations orthogonal to $\{B_+, B_-\}$ and the asymptotic distance of prototypes from the decision boundary. This corresponds to the observation that the following combinations of order parameters become proportional to η in the limit $\alpha \to \infty$:

$$r_+ = R_{++} - R_{-+}, \;\; r_- = R_{--} - R_{+-}, \;\; q = Q_{++} - Q_{--}, \;\; \delta = \sqrt{Q_{++} - 2Q_{+-} + Q_{--}} \qquad (17)$$

which results in an η-independent asymptotic generalization error, Eqs. (14,15). This finding also implies that the angle between $(w_+ - w_-)$ and $(B_+ - B_-)$ which is given by arccos $\left[(r_+ + r_-)/(\sqrt{2}\delta)\right]$ becomes independent of the learning rate for $\alpha \to \infty$.

The quantity δ in (17) measures the distance $|w_+ - w_-|$ and vanishes for $\eta \to 0$. Hence, the prototypes will coincide and the consideration of this limit together with $\alpha \to \infty$ is not useful in the case of LFM.

The mathematical analysis of the stationary state reveals another interesting feature: One can show that fixed point configurations of the system of ODE for LFM, Eq. (36), satisfy the necessary condition

$$p_+ \varepsilon_+ \; = \; p_- \varepsilon_- .$$

That is, the two contributions to the total ε_g, Eqs. (14,15), become equal for $\alpha \to \infty$. As a consequence, LFM updates will be based on balanced data, asymptotically, as they are restricted to misclassified examples.

While learning from mistakes appears to be a very plausible concept and cures the divergence of LVQ+/-, the resulting asymptotic generalization ability ε_g^{lfm} turns out rather poor. The chain lines in Figure 4 mark ε_g^{lfm} for two different model scenarios. Not only is LFM outperformed by the basic LVQ1 and LVQ+/- with early stopping, for small and large values of p_+ its generalization error can even exceed the trivial value $\min\{p_+, p_-\}$.

It is important to note that the above results apply only to the crisp (LFM) version of RSLVQ (Seo and Obermayer, 2003). It is very well possible that truly *soft* schemes display a much better generalization behavior in our model situation. In fact, it has been demonstrated that RSLVQ performs significantly better than the crisp LFM and other LVQ algorithms in practical situations (Seo and Obermayer, 2003).

4.3.2 LFM-W

Our analysis shows that the introduction of a window for the selection of data, as defined in Sec. 2.3 (d), bears the potential to improve the generalization behavior of LFM drastically. The mathematical treatment is to a large extent analogous to that of the unrestricted LFM procedure, see Appendix A for details.

Qualitatively, the typical learning curves $R_{S\tau}(\alpha), Q_{ST}(\alpha)$ of LFM-W and the corresponding projected trajectories also resemble those of unrestricted LFM. Note, however, that the stationary generalization error ε_g^{lfm-w} of LFM-W does depend on the learning rate η, in contrast to the results displayed in Figure 7 (right panel) for LFM without window.

A detailed discussion of LFM-W will be presented elsewhere. Here, we focus on the role of the window parameter δ. Figure 8 (right panel) displays the dependence of the $(\alpha \to \infty)$ asymptotic ε_g^{lfm-w} for constant learning rate $\eta = 1.0$ and example settings of the model parameters. For very large δ, the suboptimal results of unrestricted LFM are recovered and also for $\delta \to 0$ the performance deteriorates. Evidently, an optimal choice of δ exists which yields the best generalization behavior, given all other parameters. Note that the precise position of the optimum and the robustness with respect to variation of δ depend strongly on the properties of the data.

We restrict ourselves to demonstrating that the performance can improve drastically in comparison with unrestricted LFM. Figure 8 (left panel), shows the stationary generalization error as a function of the prior weight p_+ for several (fixed) window sizes in an example setting. Note that ε_g^{lfm-w} for properly chosen values of δ is significantly lower than that for unrestricted LFM, that is, $\delta \to \infty$.

The evaluation of the truly optimal generalization error in the frame of our model is beyond the scope of this publication. In a specific learning problem, that is, for a particular choice of $\lambda, v_\pm$ and $p_\pm$ in our model, the training algorithm is to be optimized with respect to the two-dimensional parameter space of η and δ. Such an optimization would be difficult to achieve in practical situations and require sophisticated validation techniques.

5. Summary and Conclusion

We have rigorously investigated several basic LVQ algorithms: the original LVQ1, LVQ+/- with and without early stopping, *learning from mistakes* (LFM), and LFM-W which includes an additional window rule. The analysis is performed by means of the theory of on-line learning in a simple though relevant model setting.

It can be seen that LVQ+/- usually displays a divergent behavior whereas LVQ1 and LFM procedures converge towards fixed positions of the order parameters. These findings correspond to observations in practical situations.

The respective convergence speed depends on the learning rate in all cases. The same holds for the quality of the resulting classifier for LVQ1, LVQ+/- with early stopping, and LFM-W. For LFM without a selective window, on the contrary, the generalization ability of the stationary setting is independent of the choice of the learning rate. It should be mentioned that the trajectories of the prototypes need not be the shortest towards the final position and, often, initial overshooting as in LVQ1 can be observed if the classes are not balanced.

Even more interesting than their dynamical behavior is the generalization ability achieved by the algorithms. LVQ1 turns out to yield surprisingly good results, not very far from the optimum achievable error for this class of problems.

The outcome of LVQ+/- with early stopping can be close to or even slightly better than the $\alpha \to \infty$ asymptotic performance of LVQ1. However, it depends strongly on initialization and the detailed properties of the data. LVQ+/- like strategies appear particularly promising for the initial phase of training or when only a very limited training set is available.

The robustness of LFM as a crisp version of RSLVQ is clearly demonstrated by the fact that its asymptotic behavior is not affected by the magnitude of the learning rate. However, quite unexpectedly, it shows rather poor generalization ability, which is even inferior to a trivial classification for highly unbalanced data. We demonstrate that a proper selection of examples close to the current decision boundary as in LFM-W can improve the generalization performance drastically. Presumably, similar improvement could be achieved by employing RSLVQ with a soft assignment as suggested by Seo and Obermayer (2003). In practical situations, both schemes would require the careful tuning of a parameter, that is, the *softness* or *window size*, in order to achieve good generalization. We will address the detailed analysis and optimization of LFM-W and RSLVQ in forthcoming studies.

A few of our results, for instance the instability of LVQ+/-, may appear plausible from elementary reasoning. Others are clearly far from obvious and demonstrate the usefulness of our systematic approach. We show, in particular, the good generalization ability of the original LVQ1 learning rule. It does not follow the gradient of a well-defined cost function but outperforms several alternative algorithms. The relatively poor performance of the highly intuitive LFM training constitutes another non-trivial insight obtained from our analysis.

Our model is currently restricted to two unimodal classes and two prototypes, a case in which the mathematical analysis is feasible but which is far away from typical settings in practice. Nevertheless, this investigation yields relevant insights into practical situations. One can expect that an algorithm which does not yield a good generalization ability in this idealized scenario is also inappropriate for more complex, practical applications. In this sense, the investigation provides a meaningful testing ground for the success of learning algorithms.

Frequently, at most two prototypes are updated at a given time step also in larger LVQ networks. Hence, such systems can be interpreted as a superposition of pairs of prototypes within classes and at class borders. Within classes, a simple vector quantization takes place, a scenario which has been investigated using the same framework (Biehl et al., 1997; Ghosh et al., 2004). At cluster borders, the situation investigated in this article is relevant. However, the role of further, complicating effects in the dynamics of the superposition of these simple subsystems has to be studied in forthcoming projects. Also, the explicit extension of the theoretical framework to multi-class, multi-prototype problems is feasible under simplifying assumptions. It is currently in the focus of our efforts.

Based on the formalism presented in this article, a variety of further research becomes possible. Naturally, the consideration of alternative LVQ schemes is necessary. Learning rules which also change the metric during training such as the one proposed by Hammer and Villmann (2002) seem particularly interesting. However, it is not obvious how highly nonlinear adaptation schemes or algorithms which single out specific data components can be treated within our theoretical framework.

The ultimate goal of this work is the design of robust and reliable LVQ algorithms which provide optimal generalization ability. One step into this direction would be the on-line optimization of algorithm parameters, for example, the learning rate, based on the observed behavior in the course of training. Note that this can be done systematically by means of an optimization of the learning curve with respect to the learning rate in our setting. Even more promising, one can formally optimize the actual update functions f_S of the learning rule with respect to the generalization ability gained per example. This should be possible along the lines of variational optimization as it has been applied in the training of perceptrons or multilayered neural networks (e.g., Saad, 1999; Engel and van den Broeck, 2001; Biehl and Caticha, 2003). An alternative could be the construction of on-line prescriptions within a Bayesian framework, as it has been employed in the context of perceptron training (for further references consult: Saad, 1999). Even if practical implementations turn out to be difficult, these investigations would help to identify the most important features of successful algorithms. Hence, this line of research should strengthen the mathematical foundation of LVQ training.

Appendix A. The Theoretical Framework

Here we outline key steps of the calculations referred to in the text. Important aspects of the formalism were first used in the context of unsupervised vector quantization (Biehl et al., 1997). Some of the calculations presented here were recently detailed in a Technical Report (Ghosh et al., 2004).

Throughout this appendix indices $l, m, k, s, \sigma \in \{\pm 1\}$ (or $\pm$ for short) represent the class labels and cluster memberships. We also use the shorthand

$$\Theta_s = \Theta\left(d_{-s} - d_{+s}\right)$$

for the Heaviside function in LVQ1. We furthermore employ the notations

$$\widehat{\Theta}^{o}_{\sigma} = \Theta\left(d_{\sigma} - d_{-\sigma}\right) \quad \text{and} \quad \widehat{\Theta}^{\delta}_{\sigma} = \Theta\left(d_{\sigma} - d_{-\sigma} - \delta\right) \tag{18}$$

for the complementary Heaviside function in LFM and the modified update of LFM-W, respectively.

A.1 Statistics of the Projections

To a large extent, our analysis is based on the observation that the projections $h_{\pm} = \mathbf{w}_{\pm} \cdot \xi$ and $b_{\pm} = \mathbf{B}_{\pm} \cdot \xi$ are correlated Gaussian random quantities for a vector ξ drawn from one of the clusters contributing to the density (1). Where convenient, we will combine the projections into a four-dimensional vector denoted as $\underline{x} = (h_+, h_-, b_+, b_-)^T$.

We will assume implicitly that ξ is statistically independent from the considered weight vectors $w_{\pm}$. This is obviously the case in our on-line prescription where the novel example ξ^{μ} is uncorrelated with all previous data and hence with $w_{\pm}^{\mu-1}$. For the sake of readability we omit indices μ in the following.

The first and second conditional moments given in Eq. (11) are obtained from the following elementary considerations.

First Moments
Exploiting the above mentioned statistical independence we can show immediately that

$$\langle h_l \rangle_k = \langle w_l \cdot \xi \rangle_k = w_l \cdot \langle \xi \rangle_k = w_l \cdot (\lambda B_k) = \lambda R_{lk}. \tag{19}$$

Similarly we get for b_l:

$$\langle b_l \rangle_k = \langle B_l \cdot \xi \rangle_k = B_l \cdot \langle \xi \rangle_k = B_l \cdot (\lambda B_k) = \lambda \delta_{lk}, \tag{20}$$

where δ_{lk} is the Kronecker delta and we exploit that B_+ and B_- are orthonormal. Now the conditional means $\underline{\mu}_k = \langle \underline{x} \rangle_k$ can be written as

$$\underline{\mu}_{k=+1} = \lambda\left(R_{++}, R_{-+}, 1, 0\right)^T \quad \text{and} \quad \underline{\mu}_{k=-1} = \lambda\left(R_{+-}, R_{--}, 0, 1\right)^T. \tag{21}$$

Second Moments

In order to compute the conditional variance or covariance $\langle h_l h_m\rangle_k - \langle h_l\rangle_k \langle h_m\rangle_k$ we first consider the average

$$\begin{aligned}
\langle h_l\, h_m\rangle_k &= \langle (w_l\cdot\xi)(w_m\cdot\xi)\rangle_k = \left\langle \left(\sum_{i=1}^N (w_l)_i(\xi)_i\right)\left(\sum_{j=1}^N (w_m)_j(\xi)_j\right)\right\rangle_k \\
&= \left\langle \sum_{i=1}^N (w_l)_i(w_m)_i(\xi)_i(\xi)_i + \sum_{i=1}^N\sum_{j=1, j\neq i}^N (w_l)_i(w_m)_j(\xi)_i(\xi)_j\right\rangle_k \\
&= \sum_{i=1}^N (w_l)_i(w_m)_i\langle(\xi)_i(\xi)_i\rangle_k + \sum_{i=1}^N\sum_{j=1, j\neq i}^N (w_l)_i(w_m)_j\langle(\xi)_i(\xi)_j\rangle_k \\
&= \sum_{i=1}^N (w_l)_i(w_m)_i[v_k + \lambda^2(B_k)_i(B_k)_i] + \sum_{i=1}^N\sum_{j=1, j\neq i}^N (w_l)_i(w_m)_j\lambda^2(B_k)_i(B_k)_j \\
&= v_k\sum_{i=1}^N (w_l)_i(w_m)_i + \lambda^2\sum_{i=1}^N (w_l)_i(w_m)_i(B_k)_i(B_k)_i \\
&\quad +\lambda^2\sum_{i=1}^N\sum_{j=1, j\neq i}^N (w_l)_i(w_m)_j(B_k)_i(B_k)_j \\
&= v_k w_l\cdot w_m + \lambda^2(w_l\cdot B_k)(w_m\cdot B_k) = v_k\, Q_{lm} + \lambda^2 R_{lk}R_{mk}\,.
\end{aligned}$$

Here we have used that components of ξ from cluster k have variance v_k and are independent. This implies for all $i,j\in\{1,\ldots,N\}$:

$$\langle(\xi)_i\,(\xi)_i\rangle_k - \langle(\xi)_i\rangle_k\,\langle(\xi)_i\rangle_k = v_k \quad\Rightarrow\quad \langle(\xi)_i\,(\xi)_i\rangle_k = v_k + \langle(\xi)_i\rangle_k\,\langle(\xi)_i\rangle_k\,,$$

and $\big\langle(\xi)_i\,(\xi)_j\big\rangle_k = \langle(\xi)_i\rangle_k\,\big\langle(\xi)_j\big\rangle_k$ for $i\neq j$.

Finally, we obtain the conditional second moment

$$\langle h_l\, h_m\rangle_k - \langle h_l\rangle_k\langle h_m\rangle_k = v_k\, Q_{lm} + \lambda^2 R_{lk}R_{mk} - \lambda^2 R_{lk}R_{mk} = v_k\, Q_{lm}\,.$$

In an analogous way we get

$$\langle b_l b_m\rangle_k - \langle b_l\rangle_k\langle b_m\rangle_k = v_k\,\delta_{lm} \quad\text{and}\quad \langle h_l b_m\rangle_k - \langle h_l\rangle_k\langle b_m\rangle_k = v_k\, R_{lm}\,.$$

The above results are summarized in Eq. (11). The conditional covariance matrix of $\underline{x}$ can be expressed explicitly in terms of the order parameters as follows:

$$C_k = v_k\begin{pmatrix} Q_{++} & Q_{+-} & R_{++} & R_{+-} \\ Q_{+-} & Q_{--} & R_{-+} & R_{--} \\ R_{++} & R_{-+} & 1 & 0 \\ R_{+-} & R_{--} & 0 & 1 \end{pmatrix}.$$

The conditional density of $\underline{x}$ for data from class k is a Gaussian $N(\underline{\mu}_k, C_k)$ where $\underline{\mu}_k$ is the conditional mean vector, Eq. (21), and C_k is given above.

A.2 Form of the Differential Equations

Here, the differential equations for $\{R_{s\sigma}, Q_{st}\}$ are given for LVQ1, LVQ+/-, LFM, and LFM-W, before Gaussian averages are performed.

A.2.1 LVQ1

In the case of the LVQ1 algorithm the generic form of the coupled system of ODE (Eq. (12)) yields the following system:

$$\begin{aligned}
\frac{dR_{lm}}{d\alpha} &= \eta\left(l\Big(\langle\sigma b_m\Theta_l\rangle - \langle\sigma\Theta_l\rangle R_{lm}\Big)\right), \\
\frac{dQ_{lm}}{d\alpha} &= \eta\left(l\left(\langle\sigma h_m\Theta_l\rangle - \langle\sigma\Theta_l\rangle Q_{lm}\right) + m\left(\langle\sigma h_l\Theta_m\rangle - \langle\sigma\Theta_m\rangle Q_{lm}\right)\right. \\
&\qquad \left. + \eta\,\delta_{lm}\sum_{\sigma=\pm1} v_\sigma p_\sigma\langle\Theta_l\rangle_\sigma\right). \qquad (22)
\end{aligned}$$

A.2.2 LVQ+/-

In the case of LVQ+/- Eq. (12) results in the following system of coupled ODE:

$$\begin{aligned}
\frac{dR_{lm}}{d\alpha} &= \eta\, l\left(\langle\sigma b_m\rangle - \langle\sigma\rangle R_{lm}\right), \\
\frac{dQ_{lm}}{d\alpha} &= \eta\left(l\langle\sigma h_m\rangle - l\langle\sigma\rangle Q_{lm} + m\langle\sigma h_l\rangle - m\langle\sigma\rangle Q_{lm} + \eta\, l\, m\sum_{\sigma=\pm1} p_\sigma v_\sigma\right). \qquad (23)
\end{aligned}$$

A.2.3 LFM

With the modulation function for LFM, Eq. (12), and using $\widehat{\Theta}^o_\sigma$ from Eq. (18) we obtain

$$\begin{aligned}
\frac{dR_{lm}}{d\alpha} &= \eta\, l\left(\langle\sigma b_m\widehat{\Theta}^o_\sigma\rangle - \langle\sigma\widehat{\Theta}^o_\sigma\rangle R_{lm}\right), \qquad (24) \\
\frac{dQ_{lm}}{d\alpha} &= \eta\left(l\langle\sigma h_m\widehat{\Theta}^o_\sigma\rangle - (l+m)\langle\sigma\widehat{\Theta}^o_\sigma\rangle Q_{lm} + m\langle\sigma h_l\widehat{\Theta}^o_\sigma\rangle + l\, m\eta\sum_{\sigma=\pm1} p_\sigma v_\sigma\langle\widehat{\Theta}^o_\sigma\rangle_\sigma\right).
\end{aligned}$$

A.2.4 LFM-W

For the modulation function of the LFM-W algorithm in Eq. (12) we obtain

$$\begin{aligned}
\frac{dR_{lm}}{d\alpha} &= \eta\, l\left(\langle\sigma b_m(\widehat{\Theta}^o_\sigma - \widehat{\Theta}^\delta_\sigma)\rangle - \langle\sigma(\widehat{\Theta}^o_\sigma - \widehat{\Theta}^\delta_\sigma)\rangle R_{lm}\right), \qquad (25) \\
\frac{dQ_{lm}}{d\alpha} &= \eta\Big(l\langle\sigma h_m(\widehat{\Theta}^o_\sigma - \widehat{\Theta}^\delta_\sigma)\rangle - (l+m)\langle\sigma(\widehat{\Theta}^o_\sigma - \widehat{\Theta}^\delta_\sigma)\rangle Q_{lm} + m\langle\sigma h_l(\widehat{\Theta}^o_\sigma - \widehat{\Theta}^\delta_\sigma)\rangle
\end{aligned}$$

$$+l\,m\eta \sum_{\sigma=\pm1} p_\sigma v_\sigma \langle(\widehat{\Theta}^o_\sigma - \widehat{\Theta}^\delta_\sigma)\rangle_\sigma \Bigg),$$

where $\widehat{\Theta}^o_\sigma$ and $\widehat{\Theta}^\delta_\sigma$ are defined in Eq. (18). Note that for $\delta \to \infty$ we recover Eqs. (24) for LFM as $\lim_{\delta\to\infty} \widehat{\Theta}^\delta = 0$.

A.3 Gaussian Averages

In order to obtain the actual ODE for a given modulation function, averages over the joint density $P(h_+, h_-, b_+, b_-)$ are performed for LVQ1, LVQ+/-, and LFM.

A.3.1 LVQ+/-

The elementary averages in Eq. (23) are directly obtained from Eqs. (19,20) and read:

$$\langle \sigma b_m \rangle = \sum_{\sigma=\pm1} \sigma p_\sigma \lambda \delta_{m,\sigma}, \qquad \langle \sigma h_m \rangle = \sum_{\sigma=\pm1} \sigma p_\sigma \lambda R_{m,\sigma}, \qquad \langle \sigma \rangle = \sum_{\sigma=\pm1} \sigma p_\sigma. \tag{26}$$

A.3.2 LVQ1

In the systems of ODE presented in Eq. (22) we encounter Heaviside functions of the following generic form:

$$\Theta_s = \Theta(\underline{\alpha}_s . \underline{x} - \beta_s)$$

which gives $\Theta_s = \Theta\left(d_{-s} - d_{+s}\right) = \Theta(\underline{\alpha}_s . \underline{x} - \beta_s)$ with

$$\underline{\alpha}_s = (+2s, -2s, 0, 0)^T \text{ and } \beta_s = \left(Q_{+s+s} - Q_{-s-s}\right), \tag{27}$$

Performing the averages in Eqs. (22) involves conditional means of the form

$$\langle(\underline{x})_n \Theta_s\rangle_k \quad \text{and} \quad \langle\Theta_s\rangle_k$$

where $(\underline{x})_n$ is the n^{th} component of $\underline{x} = (h_{+1}, h_{-1}, b_{+1}, b_{-1})$. We first address the term

$$\begin{aligned}\langle(\underline{x})_n \Theta_s\rangle_k &= \frac{(2\pi)^{-2}}{(det(C_k))^{\frac{1}{2}}} \int_{\mathbb{R}^4} (\underline{x})_n \Theta\left(\underline{\alpha}_s \cdot \underline{x} - \beta_s\right) \exp\left(-\frac{1}{2}\left(\underline{x} - \underline{\mu}_k\right)^T C_k^{-1} \left(\underline{x} - \underline{\mu}_k\right)\right) d\underline{x} \\ &= \frac{(2\pi)^{-2}}{(det(C_k))^{\frac{1}{2}}} \int_{\mathbb{R}^4} \left(\underline{x}' + \underline{\mu}_k\right)_n \Theta\left(\underline{\alpha}_s \cdot \underline{x}' + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s\right) \exp\left(-\frac{1}{2} \underline{x}'^T C_k^{-1} \underline{x}'\right) d\underline{x}'\end{aligned}$$

with the substitution $\underline{x}' = \underline{x} - \underline{\mu}_k$.

Let $\underline{x}' = C_k{}^{\frac{1}{2}} \underline{y}$, where $C_k{}^{\frac{1}{2}}$ is defined in the following way: $C_k = C_k{}^{\frac{1}{2}} C_k{}^{\frac{1}{2}}$. Since C_k is a covariance matrix, it is positive semidefinite and $C_k{}^{\frac{1}{2}}$ exists. Hence we have $d\underline{x}' = det(C_k^{\frac{1}{2}}) d\underline{y} = (det(C_k))^{\frac{1}{2}} d\underline{y}$

and
$\langle (\underline{x})_n \Theta_s \rangle_k =$

$$\begin{aligned}
&= \frac{1}{(2\pi)^2} \int_{\mathbb{R}^4} (C_k^{\frac{1}{2}} \underline{y})_n \Theta\left(\underline{\alpha}_s \cdot C_k^{\frac{1}{2}} \underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s \right) \exp\left(-\frac{1}{2} \underline{y}^2 \right) d\underline{y} + (\underline{\mu}_k)_n \langle \Theta_s \rangle_k \\
&= \frac{1}{(2\pi)^2} \int_{\mathbb{R}^4} \sum_{j=1}^{4} \left((C_k^{\frac{1}{2}})_{nj} (\underline{y})_j \right) \Theta\left(\underline{\alpha}_s \cdot C_k^{\frac{1}{2}} \underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s \right) \exp\left(-\frac{1}{2} \underline{y}^2 \right) d\underline{y} + (\underline{\mu}_k)_n \langle \Theta_s \rangle_k \\
&= I + (\underline{\mu}_k)_n \langle \Theta_s \rangle_k \qquad \text{(introducing the abbreviation } I\text{).} \qquad (28)
\end{aligned}$$

Now consider the integrals contributing to I:

$$I_j = \int_{\mathbb{R}} (C_k^{\frac{1}{2}})_{nj} (\underline{y})_j \Theta\left(\underline{\alpha}_s \cdot C_k^{\frac{1}{2}} \underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s \right) \exp\left(-\frac{1}{2} (\underline{y})_j^2 \right) d(\underline{y})_j .$$

We can perform an integration by parts, $\int u dv = uv - \int v du$, with
$u = \Theta\left(\underline{\alpha}_s \cdot C_k^{\frac{1}{2}} \underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s \right), \qquad v = (C_k^{\frac{1}{2}})_{nj} \exp\left(-\frac{1}{2} (\underline{y})_j^2 \right)$
$du = \frac{\partial}{\partial (\underline{y})_j} \Theta\left(\underline{\alpha}_s \cdot C_k^{\frac{1}{2}} \underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s \right) d(\underline{y})_j$
$dv = (-)(C_k^{\frac{1}{2}})_{nj} (\underline{y})_j \exp\left(-\frac{1}{2} (\underline{y})_j^2 \right) d(\underline{y})_j$, and obtain

$$\begin{aligned}
I_j &= -\underbrace{\left[\Theta\left(\underline{\alpha}_s \cdot C_k^{\frac{1}{2}} \underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s \right) (C_k^{\frac{1}{2}})_{nj} \exp\left(-\frac{1}{2} (\underline{y})_j^2 \right) \right]_{-\infty}^{\infty}}_{0} \\
&\quad + \left[\int_{\mathbb{R}} (C_k^{\frac{1}{2}})_{nj} \frac{\partial}{\partial (\underline{y})_j} \left(\Theta\left(\underline{\alpha}_s \cdot C_k^{\frac{1}{2}} \underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s \right) \right) \exp\left(-\frac{1}{2} (\underline{y})_j^2 \right) d(\underline{y})_j \right] \\
&= \int_{\mathbb{R}} (C_k^{\frac{1}{2}})_{nj} \frac{\partial}{\partial (\underline{y})_j} \left(\Theta\left(\underline{\alpha}_s \cdot C_k^{\frac{1}{2}} \underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s \right) \right) \exp\left(-\frac{1}{2} (\underline{y})_j^2 \right) d(\underline{y})_j .
\end{aligned}$$

In total we get

$$\begin{aligned}
I &= \frac{1}{(2\pi)^2} \sum_{j=1}^{4} (C_k^{\frac{1}{2}})_{nj} \int_{\mathbb{R}^4} \frac{\partial}{\partial (\underline{y})_j} \left(\Theta\left(\underline{\alpha}_s \cdot C_k^{\frac{1}{2}} \underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s \right) \right) \exp\left(-\frac{1}{2} \underline{y}^2 \right) d\underline{y} \\
&= \frac{1}{(2\pi)^2} \sum_{j=1}^{4} \left((C_k^{\frac{1}{2}})_{nj} \sum_{i=1}^{4} (\underline{\alpha}_s)_i (C_k^{\frac{1}{2}})_{i,j} \right) \int_{\mathbb{R}^4} \delta\left(\underline{\alpha}_s \dot{C}_k^{\frac{1}{2}} \underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s \right) \exp\left(-\frac{1}{2} \underline{y}^2 \right) d\underline{y}. \\
&= \frac{1}{(2\pi)^2} (C_k \underline{\alpha}_s)_n \int_{\mathbb{R}^4} \left(\delta\left(\underline{\alpha}_s \cdot C_k^{\frac{1}{2}} \underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s \right) \right) \exp\left(-\frac{1}{2} \underline{y}^2 \right) d\underline{y}.
\end{aligned}$$

In the last step we have used

$$\frac{\partial}{\partial (\underline{y})_j}\Theta\left(\underline{\alpha}_s \cdot C_k^{\frac{1}{2}}\underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s\right) = \sum_{i=1}^{4} (\underline{\alpha}_s)_i (C_k^{\frac{1}{2}})_{i,j} \delta(\underline{\alpha}_s \cdot C_k^{\frac{1}{2}}\underline{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s)$$

with the Dirac delta-function $\delta(.)$.

Now, note that $\exp\left[-\frac{1}{2}\underline{y}^2\right] d\underline{y}$ is a measure which is invariant under rotation of the coordinate axes. We rotate the system in such a way that one of the axes, say $\widetilde{y}$, is aligned with the vector $C_k^{\frac{1}{2}}\underline{\alpha}_s$. The remaining three coordinates can be integrated over and we get

$$I = \frac{1}{\sqrt{2\pi}}(C_k\underline{\alpha}_s)_n \int_{\mathbb{R}} \delta\left(\|C_k^{\frac{1}{2}}\underline{\alpha}_s\|\widetilde{y} + \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s\right) \exp\left[-\frac{1}{2}\widetilde{y}^2\right] d\widetilde{y}.$$

We define

$$\widetilde{\alpha}_{sk} = \|C_k^{\frac{1}{2}}\underline{\alpha}_s\| = \sqrt{\underline{\alpha}_s \cdot C_k \underline{\alpha}_s} \quad \text{and} \quad \widetilde{\beta}_{sk} = \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s \tag{29}$$

and obtain

$$\begin{aligned} I &= \frac{1}{\sqrt{2\pi}}(C_k\underline{\alpha}_s)_n \int_{\mathbb{R}} \delta\left(\widetilde{\alpha}_{sk}\widetilde{y} + \widetilde{\beta}_{sk}\right) \exp\left[-\frac{1}{2}\widetilde{y}^2\right] d\widetilde{y} \\ &= \frac{(C_k\underline{\alpha}_s)_n}{\sqrt{2\pi}\widetilde{\alpha}_{sk}} \int_{\mathbb{R}} \delta\left(z + \widetilde{\beta}_{sk}\right) \exp\left[-\frac{1}{2}\left(\frac{z}{\widetilde{\alpha}_{sk}}\right)^2\right] dz \quad (\text{with} \quad z = \alpha_{sk}\widetilde{y}) \\ &= \frac{(C_k\underline{\alpha}_s)_n}{\sqrt{2\pi}\widetilde{\alpha}_{sk}} \exp\left[-\frac{1}{2}\left(\frac{\widetilde{\beta}_{sk}}{\widetilde{\alpha}_{sk}}\right)^2\right]. \end{aligned} \tag{30}$$

Now we compute the remaining average in (28) in an analogous way and get

$$\begin{aligned} \langle \Theta_s \rangle_k &= \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} \Theta\left(\widetilde{\alpha}_{sk}\widetilde{y} + \widetilde{\beta}_{sk}\right) \exp\left[-\frac{1}{2}\widetilde{y}^2\right] d\widetilde{y} \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\frac{\widetilde{\beta}_{sk}}{\widetilde{\alpha}_{sk}}} \exp\left[-\frac{1}{2}\widetilde{y}^2\right] d\widetilde{y} = \Phi\left(\frac{\widetilde{\beta}_{sk}}{\widetilde{\alpha}_{sk}}\right) \quad \text{with} \quad \Phi(z) = \int_{-\infty}^{z} dx \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}}. \end{aligned} \tag{31}$$

Finally we obtain the required average using (30) and (31) as follows:

$$\langle (\underline{x})_n \Theta_s \rangle_k = \frac{(C_k\underline{\alpha}_s)_n}{\sqrt{2\pi}\widetilde{\alpha}_{sk}} \exp\left[-\frac{1}{2}\left(\frac{\widetilde{\beta}_{sk}}{\widetilde{\alpha}_{sk}}\right)^2\right] + (\underline{\mu}_k)_n \Phi\left(\frac{\widetilde{\beta}_{sk}}{\widetilde{\alpha}_{sk}}\right). \tag{32}$$

The quantities $\widetilde{\alpha}_{sk}$ and $\widetilde{\beta}_{sk}$ are defined through Eq. (29) and (27) for LVQ1.

A.3.3 LFM AND LFM-W

In Eqs. (24,25) we have to evaluate conditional means of the form

$$\langle (\underline{x})_n \widehat{\Theta}_s^{\delta} \rangle_k, \ \langle \widehat{\Theta}_s^{\delta} \rangle_k \ \text{with the special cases} \ \langle (\underline{x})_n \widehat{\Theta}_s^{o} \rangle_k, \ \langle \widehat{\Theta}_s^{o} \rangle_k.$$

Here we use the notation

$$\begin{aligned}\widehat{\Theta}^{\delta}_{\sigma} &= \Theta(d_\sigma - d_{-\sigma} - \delta) = \Theta(\underline{\widehat{\alpha}}_\sigma . \underline{x} - \widehat{\beta}^{\delta}_{\sigma})\\ \widehat{\Theta}^{o}_{\sigma} &= \Theta(d_\sigma - d_{-\sigma}) \qquad = \Theta(\underline{\widehat{\alpha}}_\sigma . \underline{x} - \widehat{\beta}^{o}_{\sigma})\end{aligned}$$

with $\underline{\widehat{\alpha}}_\sigma = (-2\sigma, +2\sigma, 0, 0)^T$, $\widehat{\beta}^{\delta}_{\sigma} = -(Q_{\sigma\sigma} - Q_{-\sigma-\sigma} - \delta)$, and $\widehat{\beta}^{o}_{\sigma} = -(Q_{\sigma\sigma} - Q_{-\sigma-\sigma})$.

The mathematical structure is completely analogous to the case of LVQ1 and we obtain the results

$$\langle \widehat{\Theta}^{\delta}_{s} \rangle_k = \Phi\left(\frac{\widetilde{\beta}^{\delta}_{sk}}{\widehat{\alpha}_{sk}}\right) \text{ and } \langle (\underline{x})_n \widehat{\Theta}^{\delta}_{s} \rangle_k = \frac{(C_k \underline{\widehat{\alpha}}_s)_n}{\sqrt{2\pi}\widehat{\alpha}_{sk}} \exp\left[-\frac{1}{2}\left(\frac{\widetilde{\beta}^{\delta}_{sk}}{\widehat{\alpha}_{sk}}\right)^2\right] + (\underline{\mu}_k)_n \Phi\left(\frac{\widetilde{\beta}^{\delta}_{sk}}{\widehat{\alpha}_{sk}}\right), \tag{33}$$

as well as the corresponding special cases with $\delta = 0$. Here, we have introduced

$$\widetilde{\beta}^{\delta}_{sk} = \underline{\widehat{\alpha}}_s \cdot \underline{\mu}_k - \widehat{\beta}^{\delta}_{s}, \;\; \widetilde{\beta}^{o}_{sk} = \underline{\widehat{\alpha}}_s \cdot \underline{\mu}_k - \widehat{\beta}^{o}_{s}, \;\text{ and }\; \widehat{\alpha}_{sk} = \sqrt{\underline{\widehat{\alpha}}_s \cdot C_k \underline{\widehat{\alpha}}_s}.$$

A.4 Final Form of the Differential Equations

The full form of the ODE for LVQ1, LVQ+/-, and LFM is obtained after inserting the averages given in the previous section.

A.4.1 LVQ1

For the LVQ1 algorithm, using (31) and (32), the system or ODE reads:

$$\begin{aligned}\frac{dR_{lm}}{d\alpha} &= \eta\Bigg[l\Bigg(\sum_{\sigma=\pm1}\sigma p_\sigma\Bigg[\frac{(C\underline{\alpha}_l)_{n_{bm}}}{\sqrt{2\pi}\widetilde{\alpha}_{l\sigma}}\exp\Big[-\frac{1}{2}\Big(\frac{\widetilde{\beta}_{l\sigma}}{\widetilde{\alpha}_{l\sigma}}\Big)^2\Big]\\ &\qquad +(\underline{\mu}_\sigma)_{n_{bm}}\Phi\Big(\frac{\widetilde{\beta}_{l\sigma}}{\widetilde{\alpha}_{l\sigma}}\Big)\Bigg] - \sum_{\sigma=\pm1}\sigma p_\sigma \Phi\Big(\frac{\widetilde{\beta}_{l\sigma}}{\widetilde{\alpha}_{l\sigma}}\Big)R_{lm}\Bigg)\Bigg],\\ \frac{dQ_{lm}}{d\alpha} &= \eta\Bigg(l\sum_{\sigma=\pm1}\sigma p_\sigma\Big[\frac{(C\underline{\alpha}_l)_{n_{hm}}}{\sqrt{2\pi}\widetilde{\alpha}_{l\sigma}}\exp\Big[-\frac{1}{2}\Big(\frac{\widetilde{\beta}_{l\sigma}}{\widetilde{\alpha}_{l\sigma}}\Big)^2\Big] + (\underline{\mu}_\sigma)_{n_{hm}}\Phi\Big(\frac{\widetilde{\beta}_{l\sigma}}{\widetilde{\alpha}_{l\sigma}}\Big)\Big]\\ &\qquad -l\sum_{\sigma=\pm1}\sigma p_\sigma\Big[\Phi\Big(\frac{\widetilde{\beta}_{l\sigma}}{\widetilde{\alpha}_{l\sigma}}\Big)Q_{lm} + m\sum_{\sigma=\pm1}\sigma p_\sigma\Big[\frac{(C\underline{\alpha}_l)_{n_{hl}}}{\sqrt{2\pi}\widetilde{\alpha}_{m\sigma}}\exp\Big[-\frac{1}{2}\Big(\frac{\widetilde{\beta}_{m\sigma}}{\widetilde{\alpha}_{m\sigma}}\Big)^2\Big]\\ &\qquad +(\underline{\mu}_\sigma)_{n_{hl}}\Phi\Big(\frac{\widetilde{\beta}_{m\sigma}}{\widetilde{\alpha}_{m\sigma}}\Big)\Big] - m\sum_{\sigma=\pm1}\sigma p_\sigma\Phi\Big(\frac{\widetilde{\beta}_{m\sigma}}{\widetilde{\alpha}_{m\sigma}}\Big)Q_{lm}\Bigg)\\ &\qquad +\delta_{lm}\eta^2\sum_{\sigma=\pm1}\sigma v_\sigma p_\sigma \Phi\Big(\frac{\widetilde{\beta}_{l\sigma}}{\widetilde{\alpha}_{l\sigma}}\Big).\end{aligned} \tag{34}$$

Here, we use the previously defined abbreviations, Eq. (29),

$$\widetilde{\alpha}_{sk} = \sqrt{\underline{\alpha}_s \cdot C_k \underline{\alpha}_s}, \qquad \widetilde{\beta}_{sk} = \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s$$

with $\quad \underline{\alpha}_s = (+2s, -2s, 0, 0)^T$ and $\beta_s = (Q_{+s+s} - Q_{-s-s})$.

Furthermore, $n_{hm} = \begin{cases} 1 & \text{if } m = 1\\ 2 & \text{if } m = -1\end{cases}$ and $n_{bm} = \begin{cases} 3 & \text{if } m = 1\\ 4 & \text{if } m = -1\end{cases}$.

A.4.2 LVQ+/-

Using the averages computed in (26) we get the final form of the system of ODE for the LVQ+/- algorithm as follows:

$$\begin{aligned}
\frac{dR_{lm}}{d\alpha} &= \eta\, l \left(\sum_{\sigma=\pm1} \sigma p_\sigma \lambda \delta_{m,\sigma} - \sum_{\sigma=\pm1} \sigma p_\sigma R_{lm} \right), \\
\frac{dQ_{lm}}{d\alpha} &= \eta \Bigg(l \sum_{\sigma=\pm1} \sigma p_\sigma \lambda R_{m,\sigma} - l \sum_{\sigma=\pm1} \sigma p_\sigma Q_{lm} + m \sum_{\sigma=\pm1} \sigma p_\sigma \lambda R_{l,\sigma} \\
&\qquad - m \sum_{\sigma=\pm1} \sigma p_\sigma Q_{lm} + \eta\, l\, m \sum_{\sigma=\pm1} p_\sigma v_\sigma \Bigg). \qquad (35)
\end{aligned}$$

A.4.3 LFM

The final form of the system of ODE reads with Eq. (33)

$$\begin{aligned}
\frac{dR_{lm}}{d\alpha} &= \eta\, l \Bigg(\sum_{\sigma=\pm1} \sigma p_\sigma \Big[\frac{(C\underline{\widehat{\alpha}}_\sigma)_{n_{bm}}}{\sqrt{2\pi}\widehat{\alpha}_{\sigma\sigma}} \exp\Big[-\frac{1}{2} \Big(\frac{\widetilde{\beta}^o_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}} \Big)^2 \Big] \\
&\qquad + (\underline{\mu}_\sigma)_{n_{bm}} \Phi\Big(\frac{\widetilde{\beta}^o_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}} \Big) \Big] - \sum_{\sigma=\pm1} \sigma p_\sigma \Big[\Phi\Big(\frac{\widetilde{\beta}^o_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}} \Big) \Big] R_{lm} \Bigg), \\
\frac{dQ_{lm}}{d\alpha} &= \eta \Bigg(l \sum_{\sigma=\pm1} \sigma p_\sigma \Big[\frac{(C\underline{\widehat{\alpha}}_\sigma)_{n_{hm}}}{\sqrt{2\pi}\widehat{\alpha}_{\sigma\sigma}} \exp\Big[-\frac{1}{2} \Big(\frac{\widetilde{\beta}^o_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}} \Big)^2 \Big] + (\underline{\mu}_\sigma)_{n_{hm}} \Phi\Big(\frac{\widetilde{\beta}^o_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}} \Big) \Big] \\
&\qquad - l \sum_{\sigma=\pm1} \sigma p_\sigma \Big[\Phi\Big(\frac{\widetilde{\beta}^o_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}} \Big) \Big] Q_{lm} + m \sum_{\sigma=\pm1} \sigma p_\sigma \Big[\frac{(C\underline{\widehat{\alpha}}_\sigma)_{n_{hl}}}{\sqrt{2\pi}\widehat{\alpha}_{\sigma\sigma}} \exp\Big[-\frac{1}{2} \Big(\frac{\widetilde{\beta}^o_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}} \Big)^2 \Big] \\
&\qquad + (\underline{\mu}_\sigma)_{n_{hl}} \Phi\Big(\frac{\widetilde{\beta}^o_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}} \Big) \Big] - m \sum_{\sigma=\pm1} \sigma p_\sigma \Big[\Phi\Big(\frac{\widetilde{\beta}^o_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}} \Big) \Big] Q_{lm} + l\, m\, \eta \sum_{\sigma=\pm1} v_\sigma p_\sigma \Phi\Big(\frac{\widetilde{\beta}^o_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}} \Big) \Bigg).
\end{aligned}$$
$$(36)$$

Here we have to insert $\widehat{\alpha}_{sk} = \sqrt{\underline{\widehat{\alpha}}_s \cdot C_k \underline{\widehat{\alpha}}_s}, \quad \widetilde{\beta}^o_{sk} = \underline{\widehat{\alpha}}_s \cdot \underline{\mu}_k - \widehat{\beta}^o_s$

with $\quad \underline{\widehat{\alpha}}_\sigma = (-2\sigma, +2\sigma, 0, 0)^T$ and $\widehat{\beta}^o_\sigma = -(Q_{+\sigma+\sigma} - Q_{-\sigma-\sigma})$.

Also, $\quad n_{hm} = \begin{cases} 1 & \text{if } m = 1 \\ 2 & \text{if } m = -1 \end{cases}$ and $n_{bm} = \begin{cases} 3 & \text{if } m = 1 \\ 4 & \text{if } m = -1 \end{cases}$.

A.4.4 LFM-W

The system of ODE for LFM-W, using Eq. (33), is given by

$$\frac{dR_{lm}}{d\alpha} = \eta\, l \Bigg(\sum_{\sigma=\pm1} \sigma p_\sigma \Big[\frac{(C\underline{\widehat{\alpha}}_\sigma)_{n_{bm}}}{\sqrt{2\pi}\widehat{\alpha}_{\sigma\sigma}} \exp\Big[-\frac{1}{2} \Big(\frac{\widetilde{\beta}^o_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}} \Big)^2 \Big]$$

$$
\begin{aligned}
&+(\underline{\mu}_{\sigma})_{n_{bm}}\Phi\Big(\frac{\widetilde{\beta}^{o}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Big] - \sum_{\sigma=\pm1}\sigma p_\sigma\Big[\Phi\Big(\frac{\widetilde{\beta}^{o}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Big]R_{lm}\Bigg)\\
&-\eta\, l\Bigg(\sum_{\sigma=\pm1}\sigma p_\sigma\Big[\frac{(C\underline{\widehat{\alpha}}_\sigma)_{n_{bm}}}{\sqrt{2\pi}\widehat{\alpha}_{\sigma\sigma}}\exp\Big[-\frac{1}{2}\Big(\frac{\widetilde{\beta}^{\delta}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)^2\Big]\\
&+(\underline{\mu}_{\sigma})_{n_{bm}}\Phi\Big(\frac{\widetilde{\beta}^{\delta}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Big] - \sum_{\sigma=\pm1}\sigma p_\sigma\Big[\Phi\Big(\frac{\widetilde{\beta}^{\delta}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Big]R_{lm}\Bigg),
\end{aligned}
$$

$$
\begin{aligned}
\frac{dQ_{lm}}{d\alpha} = \;& \eta\Bigg(l\sum_{\sigma=\pm1}\sigma p_\sigma\Big[\frac{(C\underline{\widehat{\alpha}}_\sigma)_{n_{hm}}}{\sqrt{2\pi}\widehat{\alpha}_{\sigma\sigma}}\exp\Big[-\frac{1}{2}\Big(\frac{\widetilde{\beta}^{o}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)^2\Big]+(\underline{\mu}_{\sigma})_{n_{hm}}\Phi\Big(\frac{\widetilde{\beta}^{o}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Big]\\
&-l\sum_{\sigma=\pm1}\sigma p_\sigma\Big[\Phi\Big(\frac{\widetilde{\beta}^{o}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Big]Q_{lm}+m\sum_{\sigma=\pm1}\sigma p_\sigma\Big[\frac{(C\underline{\widehat{\alpha}}_\sigma)_{n_{hl}}}{\sqrt{2\pi}\widehat{\alpha}_{\sigma\sigma}}\exp\Big[-\frac{1}{2}\Big(\frac{\widetilde{\beta}^{o}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)^2\Big]\\
&+(\underline{\mu}_{\sigma})_{n_{hl}}\Phi\Big(\frac{\widetilde{\beta}^{o}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Big]-m\sum_{\sigma=\pm1}\sigma p_\sigma\Big[\Phi\Big(\frac{\widetilde{\beta}^{o}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Big]Q_{lm}+l\,m\,\eta\sum_{\sigma=\pm1}v_\sigma p_\sigma\Phi\Big(\frac{\widetilde{\beta}^{o}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Bigg)\\
&-\eta\Bigg(l\sum_{\sigma=\pm1}\sigma p_\sigma\Big[\frac{(C\underline{\widehat{\alpha}}_\sigma)_{n_{hm}}}{\sqrt{2\pi}\widehat{\alpha}_{\sigma\sigma}}\exp\Big[-\frac{1}{2}\Big(\frac{\widetilde{\beta}^{\delta}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)^2\Big]+(\underline{\mu}_{\sigma})_{n_{hm}}\Phi\Big(\frac{\widetilde{\beta}^{\delta}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Big]\\
&-l\sum_{\sigma=\pm1}\sigma p_\sigma\Big[\Phi\Big(\frac{\widetilde{\beta}^{\delta}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Big]Q_{lm}+m\sum_{\sigma=\pm1}\sigma p_\sigma\Big[\frac{(C\underline{\widehat{\alpha}}_\sigma)_{n_{hl}}}{\sqrt{2\pi}\widehat{\alpha}_{\sigma\sigma}}\exp\Big[-\frac{1}{2}\Big(\frac{\widetilde{\beta}^{\delta}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)^2\Big]\\
&+(\underline{\mu}_{\sigma})_{n_{hl}}\Phi\Big(\frac{\widetilde{\beta}^{\delta}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Big]-m\sum_{\sigma=\pm1}\sigma p_\sigma\Big[\Phi\Big(\frac{\widetilde{\beta}^{\delta}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Big]Q_{lm}+l\,m\,\eta\sum_{\sigma=\pm1}v_\sigma p_\sigma\Phi\Big(\frac{\widetilde{\beta}^{\delta}_{\sigma\sigma}}{\widehat{\alpha}_{\sigma\sigma}}\Big)\Bigg).
\end{aligned}
$$

Here again, we have to insert

$$\widehat{\alpha}_{sk} = \sqrt{\underline{\widehat{\alpha}}_s\cdot C_k\,\underline{\alpha}_s},\quad \widetilde{\beta}^{\delta}_{sk} = \underline{\widehat{\alpha}}_s\cdot\underline{\mu}_k - \widehat{\beta}^{\delta}_s,\quad \widetilde{\beta}^{o}_{sk} = \underline{\widehat{\alpha}}_s\cdot\underline{\mu}_k - \widehat{\beta}^{o}_s$$

with $\underline{\widehat{\alpha}}_\sigma = (-2\sigma,+2\sigma,0,0)^T, \widehat{\beta}^{\delta}_\sigma = -(Q_{+\sigma+\sigma}-Q_{-\sigma-\sigma}-\delta)$, and $\widehat{\beta}^{o}_\sigma = -(Q_{+\sigma+\sigma}-Q_{-\sigma-\sigma})$.

As above, $n_{hm} = \begin{cases} 1 & \text{if } m=1\\ 2 & \text{if } m=-1\end{cases}$ and $n_{bm} = \begin{cases} 3 & \text{if } m=1\\ 4 & \text{if } m=-1\end{cases}$.

A.5 Analytical Results for LVQ+/-

The system of ODE (35) can be integrated analytically and the solutions are presented below for initialization of prototypes in the origin, that is, $R_{lm}(0) = Q_{lm}(0) = 0$.
For convenience, we use the parameterization

$$p_+ = \frac{1+\widehat{p}}{2},\qquad p_- = \frac{1-\widehat{p}}{2},\quad \text{with the } \textit{bias}\ \widehat{p} = (2p_+-1)\in[-1,1].$$

In the generic case of unequal priors $\widehat{p}\neq 0$, one obtains

$$
\begin{aligned}
R_{++}(\alpha) &= -\frac{1}{2\widehat{p}}\lambda(1+\widehat{p})(e^{-\alpha\eta\widehat{p}}-1), \qquad R_{+-}(\alpha) = \frac{1}{2\widehat{p}}\lambda(1-\widehat{p})(e^{-\alpha\eta\widehat{p}}-1),\\
R_{-+}(\alpha) &= -\frac{1}{2\widehat{p}}\lambda(1+\widehat{p})(e^{+\alpha\eta\widehat{p}}-1), \qquad R_{--}(\alpha) = \frac{1}{2\widehat{p}}\lambda(1-\widehat{p})(e^{+\alpha\eta\widehat{p}}-1),
\end{aligned}
$$

$$\begin{aligned}
Q_{++}(\alpha) &= \frac{1}{4\widehat{p}^2} e^{-2\alpha\eta\widehat{p}} \Big(2(e^{\alpha\eta\widehat{p}}-1)^2 \lambda^2 (1+\widehat{p}^2) + (e^{2\alpha\eta\widehat{p}}-1)\,\eta\,\widehat{p}\,(v_+(1+\widehat{p})+v_-(1-\widehat{p})) \Big), \\
Q_{+-}(\alpha) &= \frac{1}{2\widehat{p}^2} e^{-\alpha\eta\widehat{p}} \Big(-(e^{\alpha\eta\widehat{p}}-1)^2 \lambda^2 (1+\widehat{p}^2) - \alpha e^{\alpha\eta\widehat{p}} \eta^2 \widehat{p}^2 \,(v_+(1+\widehat{p})+v_-(1-\widehat{p})) \Big), \\
Q_{--}(\alpha) &= \frac{1}{4\widehat{p}^2} \Big(2(e^{\alpha\eta\widehat{p}}-1)^2 \lambda^2 (1+\widehat{p}^2) + (e^{2\alpha\eta\widehat{p}}-1)\,\eta\,\widehat{p}\,(v_+(1+\widehat{p})+v_-(1-\widehat{p})) \Big).
\end{aligned} \tag{37}$$

The special case of equal prior probabilities is obtained in the limit $\widehat{p} \to 0$:

$$R_{lm}(\alpha) = l\,m\,\frac{\lambda\eta}{2}\,\alpha \qquad Q_{lm}(\alpha) = l\,m\,\frac{1}{2}\eta^2(\alpha\lambda^2 + v_+ + v_-)\,\alpha. \tag{38}$$

A.6 The Generalization Error

Using (31) we can directly compute the generalization error as follows:

$$\varepsilon_g = \sum_{k=\pm1} p_{-k} \langle \Theta_k \rangle_{-k} = \sum_{k=\pm1} p_{-k} \Phi\Big(\frac{\widetilde{\beta}_{k-k}}{\widetilde{\alpha}_{k-k}} \Big)$$

which yields Eqs. (14,15) in the text after inserting
$\widetilde{\alpha}_{sk} = \|C_k^{\frac{1}{2}} \underline{\alpha}_s\| = \sqrt{\underline{\alpha}_s \cdot C_k \underline{\alpha}_s}, \quad \widetilde{\beta}_{sk} = \underline{\alpha}_s \cdot \underline{\mu}_k - \beta_s$.

with $\quad \underline{\alpha}_s = (+2s, -2s, 0, 0)^T$ and $\beta_s = (Q_{+s+s} - Q_{-s-s})$.

References

N. Barkai, H.S. Seung, and H. Sompolinksy. Scaling laws in learning of classification tasks. *Physical Review Letters*, 70(20):3167–3170, 1993.

M. Biehl and N. Caticha. The statistical mechanics of on-line learning and generalization. In M.A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 1095–1098. MIT Press, Cambridge, MA, 2003.

M. Biehl, A. Freking, and G. Reents. Dynamics of on-line competitive learning. *Europhysics Letters*, 38(1):73–78, 1997.

M. Biehl, A. Ghosh, and B. Hammer. The dynamics of learning vector quantization. In M. Verleysen, editor, *European Symposium on Artificial Neural Networks, ESANN'05*, pages 13–18. d-side, Evere, Belgium, 2005.

T. Bojer, B. Hammer, and C. Koers. Monitoring technical systems with prototype based clustering. In M. Verleysen, editor, *European Symposium on Artificial Neural Networks*, pages 433–439. d-side, Evere, Belgium, 2003.

L. Bottou. Stochastic gradient learning in neural networks. In *Proc. of Neuro-Nimes 91*. EC2 editions, 1991.

K. Crammer, R. Gilad-Bachrach, A. Navot, and A. Tishby. Margin analysis of the LVQ algorithm. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, pages 462–469. MIT Press, Cambridge, MA, 2003.

R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley, New York, 2000.

A. Engel and C. van den Broeck. *The Statistical Mechanics of Learning*. Cambridge University Press, Cambridge, UK, 2001.

A. Ghosh, M. Biehl, A. Freking, and G. Reents. A theoretical framework for analysing the dynamics of LVQ: A statistical physics approach. *Technical Report 2004-9-02, Mathematics and Computing Science, University Groningen, P.O. Box 800, 9700 AV Groningen, The Netherlands, available from www.cs.rug.nl/~biehl*, 2004.

A. Ghosh, M. Biehl, and B. Hammer. Dynamical analysis of LVQ type learning rules. In M. Cottrell, editor, *Workshop on the Self-Organizing-Map WSOM'05*. Univ. de Paris (I), 2005.

B. Hammer and T. Villmann. Generalized relevance learning vector quantization. *Neural Networks*, 15:1059–1068, 2002.

B. Hammer, M. Strickert, and T. Villmann. On the generalization ability of GRLVQ networks. *Neural Processing Letters*, 21(2):109–120, 2005a.

B. Hammer, M. Strickert, and T. Villmann. Prototype based recognition of splice sites. In U. Seiffert, L. C. Jain, and P. Schweitzer, editors, *Bioinformatics using Computational Intelligence Paradigms*, pages 25–55. Springer, Berlin, 2005b.

B. Hammer, M. Strickert, and T. Villmann. Supervised neural gas with general similarity measure. *Neural Processing Letters*, 21(1):21–44, 2005c.

T. Kohonen. Learning vector quantization. In M.A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks.*, pages 537–540. MIT Press, Cambridge, MA, 1995.

T. Kohonen. *Self-Organizing Maps*. Springer, Berlin, 1997.

T. Kohonen. Improved versions of learning vector quantization. *In Proc. of the International Joint conference on Neural Networks (San Diego, 1990)*, 1:545–550, 1990.

T. Kohonen, G. Barna, and R. Chrisley. Statistical pattern recognition with neural network: Benchmarking studies. In *Proc. of the IEEE second international conference on Neural Networks (San Diego, 1988)*, volume 1, pages 61–68. IEEE, New York, 1988.

L. I. Kuncheva. Classifier ensembles for changing environments. In F. Roli, J. Kittler, and T. Windeatt, editors, *Multiple Classifier Systems: 5th International Workshop, MCS2004, Cagliari, Italy*, volume 3077 of *Lecture Notes in Computer Science*, pages 1–15. Springer, Berlin, 2004.

C. Marangi, M. Biehl, and S.A. Solla. Supervised learning from clustered input examples. *Europhysics Letters*, 30(2):117, 1995.

R. Meir. Empirical risk minimization versus maximum-likelihood estimation: a case study. *Neural Computation*, 7(1):144–157, 1995.

Neural Networks Research Centre, Helsinki. Bibliography on the self-organizing maps (SOM) and learning vector quantization (LVQ). *Otaniemi: Helsinki Univ. of Technology. Available on-line: http://liinwww.ira.uka.de/bibliography/Neural/SOM.LVQ.html*, 2002.

M. Pregenzer, G. Pfurtscheller, and D. Flotzinger. Automated feature selection with distinction sensitive learning vector quantization. *Neurocomputing*, 11:19–20, 1996.

G. Reents and R. Urbanczik. Self-averaging and on-line learning. *Physical Review Letters*, 80(24): 5445–5448, 1998.

P. Riegler, M. Biehl, S.A. Solla, and C. Marangi. On-line learning from clustered input examples. In M. Marinaro and R. Tagliaferri, editors, *Neural Nets WIRN Vietri-95, Proc. of the 7th Italian Workshop on Neural Nets*, pages 87–92. World Scientific, Singapore, 1996.

D. Saad, editor. *Online learning in neural networks*. Cambridge University Press, Cambridge, UK, 1999.

A.S. Sato and K. Yamada. Generalized learning vector quantization. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 423–429, 1995.

A.S. Sato and K. Yamada. An analysis of convergence in generalized LVQ. In L. Niklasson, M. Bodén, and T. Ziemke, editors, *International Conference on Artificial Neural Networks, ICANN'98*, pages 172–176. Springer, Berlin, 1998.

F.-M. Schleif, T. Villmann, and B. Hammer. Local metric adaptation for soft nearest prototype classification to classify proteomic data. In I. Bloch, A. Petrosino, and A.G.B. Tettamanzi, editors, *International Workshop on Fuzzy Logic and Applications*, volume 3849 of *Lecture Notes in Computer Science*, pages 290–296. Springer, Berlin, 2006.

S. Seo and K. Obermayer. Soft learning vector quantization. *Neural Computation*, 15:1589–1604, 2003.

S. Seo, M. Bode, and K. Obermayer. Soft nearest prototype classification. *IEEE Transactions on Neural Networks*, 14(2):390–398, 2003.

T. Villmann, E. Merenyi, and B. Hammer. Neural maps in remote sensing image analysis. *Neural Networks*, 16(3-4):389–403, 2003.

T. H. L. Watkin, A. Rau, and M. Biehl. The statistical mechanics of learning a rule. *Reviews of Modern Physics*, 65:499–556, 1993.

Journal of Machine Learning Research 8 (2007) 361-383 Submitted 8/06; Revised 12/06; Published 2/07

Statistical Consistency of Kernel Canonical Correlation Analysis

Kenji Fukumizu FUKUMIZU@ISM.AC.JP
Institute of Statistical Mathematics
4-6-7 Minami-Azabu, Minato-ku
Tokyo 106-8569 Japan

Francis R. Bach FRANCIS.BACH@MINES.ORG
Centre de Morphologie Mathématique
Ecole des Mines de Paris
35, rue Saint-Honoré
77300 Fontainebleau, France

Arthur Gretton ARTHUR.GRETTON@TUEBINGEN.MPG.DE
Department Schölkopf
Max Planck Institute for Biological Cybernetics
Spemannstraße 38, 72076 Tübingen, Germany

Editor: Aapo Hyvarinen

Abstract

While kernel canonical correlation analysis (CCA) has been applied in many contexts, the convergence of finite sample estimates of the associated functions to their population counterparts has not yet been established. This paper gives a mathematical proof of the statistical convergence of kernel CCA, providing a theoretical justification for the method. The proof uses covariance operators defined on reproducing kernel Hilbert spaces, and analyzes the convergence of their empirical estimates of finite rank to their population counterparts, which can have infinite rank. The result also gives a sufficient condition for convergence on the regularization coefficient involved in kernel CCA: this should decrease as $n^{-1/3}$, where n is the number of data.

Keywords: canonical correlation analysis, kernel, consistency, regularization, Hilbert space

1. Introduction

Kernel methods (Cristianini and Shawe-Taylor, 2000; Schölkopf and Smola, 2002) have recently been developed as a methodology for nonlinear data analysis with positive definite kernels. In kernel methods, data are represented as functions or elements in reproducing kernel Hilbert spaces (RKHS), which are associated with positive definite kernels. The application of various linear methods in these Hilbert spaces is possible due to the reproducing property, which makes computation of inner product in the Hilbert spaces tractable. Many methods have been proposed as nonlinear extensions of conventional linear methods, such as kernel principal component analysis (Schölkopf et al., 1998), kernel Fisher discriminant analysis (Mika et al., 1999), and so on.

Kernel canonical correlation analysis (kernel CCA) was proposed (Akaho, 2001; Melzer et al., 2001; Bach and Jordan, 2002) as a nonlinear extension of canonical correlation analysis with positive definite kernels. Given two random variables X and Y, kernel CCA aims at extracting the information which is shared by the two random variables. More precisely, the purpose of kernel

CCA is to provide nonlinear mappings $f(X)$ and $g(Y)$, where f and g belong to the respective RKHS $\mathcal{H}_X$ and $\mathcal{H}_Y$, such that their correlation is maximized. Kernel CCA has been successfully applied in practice for extracting nonlinear relations between variables in genomic data (Yamanishi et al., 2003), fMRI brain images (Hardoon et al., 2004), chaotic time series (Suetani et al., 2006) and independent component analysis (Bach and Jordan, 2002).

As in many statistical methods, the target functions defined in the population case are in practice estimated from a finite sample. Thus, the convergence of the estimated functions to the population functions with increasing sample size is very important to justify the method. Since the goal of kernel CCA is to estimate a pair of functions, the convergence should be evaluated in an appropriate functional norm; we thus need tools from functional analysis to characterize the type of convergence.

The purpose of this paper is to rigorously prove the statistical consistency of kernel CCA. In proving the consistency of kernel CCA, we show also the consistency of a pair of functions which may be used as an alternative method for expressing the nonlinear dependence of two variables. The latter method uses the eigenfunctions of a NOrmalized Cross-Covariance Operator, and we call it NOCCO for short.

Both kernel CCA and NOCCO require a regularization coefficient, which is similar to Tikhonov regularization (Groetsch, 1984), to enforce smoothness of the functions in the finite sample case (thus avoiding a trivial solution) and to enable operator inversion; but the decay of this regularization with increased sample size has not yet been established. The main theorems in this paper give a sufficient condition on the decay of the regularization coefficient for the finite sample estimators to converge to the desired functions in the population limit.

Another important issue in establishing convergence is an appropriate distance measure for functions. For NOCCO, we obtain convergence in the norm of the associated RKHS. This result is very strong: if the positive definite kernels are continuous and bounded, the norm is stronger than the uniform norm in the space of continuous functions, and thus the estimated functions converge uniformly to the desired ones. For kernel CCA, we prove convergence in the L_2 norm, which is a standard distance measure for functions.

There are earlier studies relevant to the convergence of functional correlation analysis. Among others, Breiman and Friedman (1985) propose alternating conditional expectation (ACE), an iterative algorithm for functional CCA and more general regression, and demonstrate statistical consistency of the algorithm for an infinite amount of data.

Most relevant to this paper are several studies on the consistency of CCA with positive definite kernels, notably the work on nonlinear CCA for stochastic processes by Leurgans et al. (1993); He et al. (2003), who also provide consistency results. An alternative approach is to study the eigenfunctions of the cross-covariance operators, without normalising by the variance, as in the constrained covariance (COCO, Gretton et al., 2005b). We will discuss the relation between our results and these studies.

We begin our presentation in Section 2 with a review of kernel CCA and related methods, formulating them in terms of cross-covariance operators, which are the basic tools to analyze correlation in functional spaces. In Section 3, we describe the two main theorems, which respectively show the convergence of kernel CCA and NOCCO. Section 4 contains numerical results to illustrate the behavior of the methods. Section 5 is devoted to the proof of the main theorems. Some basic facts from functional analysis and general lemmas are summarized in the Appendix.

2. Kernel Canonical Correlation Analysis

In this section, we briefly review kernel CCA, following Bach and Jordan (2002), and reformulate it with covariance operators on RKHS. For the detail of positive definite kernels and RKHS, see Aronszajn (1950).

In this paper, a Hilbert space always means a separable Hilbert space, and an operator a linear operator. The operator norm of a bounded operator T is denoted by $\|T\|$ and defined as $\|T\| = \sup_{\|f\|=1} \|Tf\|$. The null space and the range of an operator $T : \mathcal{H}_1 \to \mathcal{H}_2$ are denoted by $\mathcal{N}(T)$ and $\mathcal{R}(T)$, respectively; that is, $\mathcal{N}(T) = \{f \in \mathcal{H}_1 \mid Tf = 0\}$ and $\mathcal{R}(T) = \{Tf \in \mathcal{H}_2 \mid f \in \mathcal{H}_1\}$.

Throughout this paper, $(\mathcal{X}, \mathcal{B}_\mathcal{X})$ and $(\mathcal{Y}, \mathcal{B}_\mathcal{Y})$ are measurable spaces, and $(\mathcal{H}_\mathcal{X}, k_\mathcal{X})$ and $(\mathcal{H}_\mathcal{Y}, k_\mathcal{Y})$ are RKHS of functions on $\mathcal{X}$ and $\mathcal{Y}$, respectively, with measurable positive definite kernels $k_\mathcal{X}$ and $k_\mathcal{Y}$. We consider a random vector $(X,Y) : \Omega \to \mathcal{X} \times \mathcal{Y}$ with law P_{XY}. The marginal distributions of X and Y are denoted by P_X and P_Y, respectively. It is always assumed that the positive definite kernels satisfy

$$E_X[k_\mathcal{X}(X,X)] < \infty \quad \text{and} \quad E_Y[k_\mathcal{Y}(Y,Y)] < \infty. \tag{1}$$

Note that under this assumption $\mathcal{H}_\mathcal{X}$ and $\mathcal{H}_\mathcal{Y}$ are continuously included in $L_2(P_X)$ and $L_2(P_Y)$, respectively, where $L_2(\mu)$ denotes the Hilbert space of square integrable functions with respect to the measure μ. This is easily verified by $E_X[f(X)^2] = E_X[\langle f, k_\mathcal{X}(\cdot, X)\rangle^2] \le E_X[\|f\|^2_{\mathcal{H}_\mathcal{X}} \|k_\mathcal{X}(\cdot,X)\|^2_{\mathcal{H}_\mathcal{X}}] = \|f\|^2_{\mathcal{H}_\mathcal{X}} E_X[k_\mathcal{X}(X,X)]$ for $f \in \mathcal{H}_\mathcal{X}$.

2.1 CCA in Reproducing Kernel Hilbert Spaces

Classical CCA (e.g., Greenacre, 1984) looks for linear mappings $a^T X$ and $b^T Y$ that achieve maximum correlation. Kernel CCA extends this approach by looking for functions $f \in \mathcal{H}_\mathcal{X}$ and $g \in \mathcal{H}_\mathcal{Y}$ such that the random variables $f(X)$ and $g(Y)$ have maximal correlation. More precisely, kernel CCA solves the following problem:[1]

$$\max_{\substack{f \in \mathcal{H}_\mathcal{X}, g \in \mathcal{H}_\mathcal{Y} \\ f \neq 0, g \neq 0}} \frac{\mathrm{Cov}[f(X), g(Y)]}{\mathrm{Var}[f(X)]^{1/2} \mathrm{Var}[g(Y)]^{1/2}}. \tag{2}$$

The maximizing functions f and g are decided up to scale.

In practice, we have to estimate the desired function from a finite sample. Given an i.i.d. sample $(X_1,Y_1), \ldots, (X_n,Y_n)$ from the distribution P_{XY}, an empirical estimate of Eq. (2) is

$$\max_{\substack{f \in \mathcal{H}_\mathcal{X}, g \in \mathcal{H}_\mathcal{Y} \\ f \neq 0, g \neq 0}} \frac{\widehat{\mathrm{Cov}}[f(X), g(Y)]}{\left(\widehat{\mathrm{Var}}[f(X)] + \varepsilon_n \|f\|^2_{\mathcal{H}_\mathcal{X}}\right)^{1/2} \left(\widehat{\mathrm{Var}}[g(Y)] + \varepsilon_n \|g\|^2_{\mathcal{H}_\mathcal{Y}}\right)^{1/2}}, \tag{3}$$

1. In Eq. (2) we assume $\mathrm{Var}[f(X)] \neq 0$ and $\mathrm{Var}[g(Y)] \neq 0$. See Section 2.2 for discussion on conditions under which an RKHS includes a function leading to null variance.

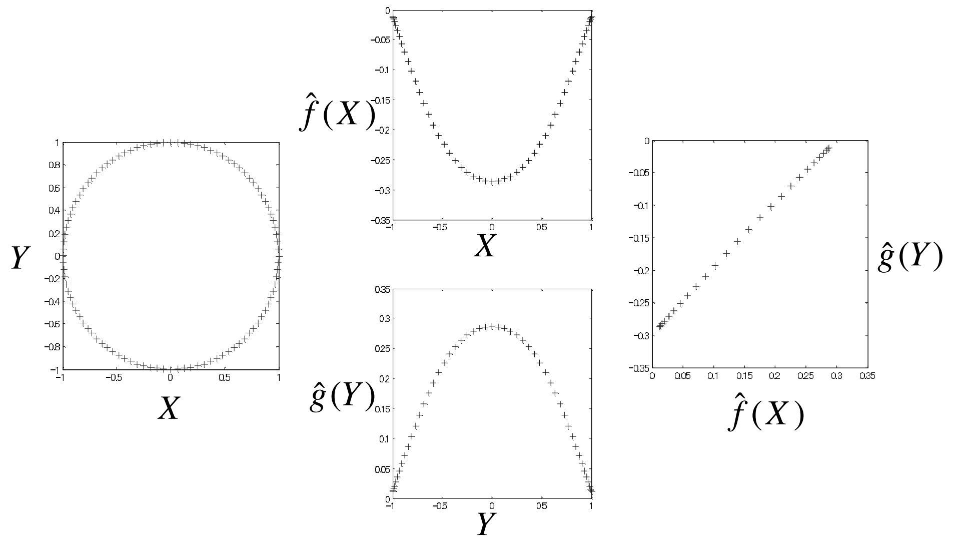


Figure 1: An example of kernel CCA. A Gaussian RBF kernel $k(x,y) = \exp\left(-\frac{1}{2\sigma^2}(x-y)^2\right)$ is used for both X and Y. Left: the original data. Center: derived functions $\widehat{f}(X_i)$ and $\widehat{g}(Y_i)$. Right: transformed data.

where

$$\widehat{\mathrm{Cov}}[f(X),g(Y)] = \frac{1}{n}\sum_{i=1}^{n}\Big(f(X_i) - \frac{1}{n}\sum_{j=1}^{n} f(X_j)\Big)\Big(g(Y_i) - \frac{1}{n}\sum_{j=1}^{n} g(Y_j)\Big),$$

$$\widehat{\mathrm{Var}}[f(X)] = \frac{1}{n}\sum_{i=1}^{n}\Big(f(X_i) - \frac{1}{n}\sum_{j=1}^{n} f(X_j)\Big)^2,$$

$$\widehat{\mathrm{Var}}[g(Y)] = \frac{1}{n}\sum_{i=1}^{n}\Big(g(Y_i) - \frac{1}{n}\textstyle\sum_{j=1}^{n} g(Y_j)\Big)^2,$$

and a positive constant ε_n is the regularization coefficient (Bach and Jordan, 2002). As we shall see, the regularization terms $\varepsilon_n\|f\|^2_{\mathcal{H}_X}$ and $\varepsilon_n\|g\|^2_{\mathcal{H}_Y}$ make the problem well-formulated statistically, enforce smoothness, and enable operator inversion, as in Tikhonov regularization (Groetsch, 1984). For this smoothing effect, see also the discussion by Leurgans et al. (1993, Section 3).

Figure 1 shows the result of kernel CCA for a synthetic data set. The nonlinear mappings clarify the strong dependency between X and Y. Note that the dependency of the original data cannot be captured by classical CCA, because they have no linear correlation.

2.2 Cross-covariance Operators on RKHS

Kernel CCA and related methods can be formulated using cross-covariance operators, which make the theoretical analysis easier. Cross-covariance operators have also been used to derive practical methods for measuring the dependence of random variables (Fukumizu et al., 2004; Gretton et al., 2005a). This subsection reviews the basic properties of cross-covariance operators. For more details, see Baker (1973), Fukumizu et al. (2004), and Gretton et al. (2005a). The *cross-covariance* *operator*[2] of (X,Y) is an operator from $\mathcal{H}_X$ to $\mathcal{H}_\mathcal{Y}$, which is defined by

$$\langle g, \Sigma_{YX} f\rangle_{\mathcal{H}_\mathcal{Y}} = E_{XY}\big[(f(X)-E_X[f(X)])(g(Y)-E_Y[g(Y)])\big] \quad (= \mathrm{Cov}[f(X),g(Y)])$$

for all $f \in \mathcal{H}_X$ and $g \in \mathcal{H}_\mathcal{Y}$. By regarding the right hand side as a linear functional on the direct product $\mathcal{H}_X \otimes \mathcal{H}_\mathcal{Y}$, Riesz's representation theorem (Reed and Simon, 1980, for example) guarantees the existence and uniqueness of a bounded operator Σ_{YX}. The cross-covariance operator expresses the covariance between functions in the RKHS as a bilinear functional, and contains all the information regarding the dependence of X and Y expressible by nonlinear functions in the RKHS.

Obviously, $\Sigma_{YX} = \Sigma_{XY}^*$, where T^* denotes the adjoint of an operator T. In particular, if Y is equal to X, the self-adjoint operator Σ_{XX} is called the *covariance operator*. Note that $f \in \mathcal{N}(\Sigma_{XX})$ if and only if $\mathrm{Var}_X[f(X)] = 0$. The null space $\mathcal{N}(\Sigma_{XX})$ is equal to $\{f \in \mathcal{H}_X \mid f(X) = \text{constant almost surely}\}$. Under the assumptions that $\mathcal{X}$ is a topological space with continuous kernel k_X and the support of P_X is $\mathcal{X}$, the null space $\mathcal{N}(\Sigma_{XX})$ is equal to $\mathcal{H}_X \cap \mathbb{R}$, where $\mathbb{R}$ denotes the constant functions. For the Gaussian RBF kernel $k(x,y) = \exp\left(-\frac{1}{2\sigma^2}\|x-y\|^2\right)$ defined on $\mathcal{X} \subset \mathbb{R}^m$, it is known (Steinwart et al., 2004) that if the interior of $\mathcal{X}$ is not empty, a nontrivial constant function is not included in the RKHS; thus $\mathcal{N}(\Sigma_{XX}) = \{0\}$ in such cases.

The *mean element* $m_X \in \mathcal{H}_X$ with respect to a random variable X is defined as

$$\langle f, m_X\rangle_{\mathcal{H}_X} = E_X[f(X)] = E_X[\langle f, k_X(\cdot,X)\rangle_{\mathcal{H}_X}] \qquad (\forall f \in \mathcal{H}_X). \tag{4}$$

The existence and uniqueness of m_X is proved again by Riesz's representation theorem. Using the mean elements, the cross-covariance operator Σ_{YX} is rewritten

$$\langle g, \Sigma_{YX} f\rangle_{\mathcal{H}_\mathcal{Y}} = E_{XY}[\langle f, k_X(\cdot,X)-m_X\rangle_{\mathcal{H}_X} \langle k_\mathcal{Y}(\cdot,Y)-m_Y, g\rangle_{\mathcal{H}_\mathcal{Y}}].$$

Let $(X_1,Y_1),\ldots,(X_n,Y_n)$ be i.i.d. random vectors on $\mathcal{X}\times\mathcal{Y}$ with distribution P_{XY}. The *empirical cross-covariance operator* $\widehat{\Sigma}_{YX}^{(n)}$ is defined as the cross-covariance operator with the empirical distribution $\frac{1}{n}\sum_{i=1}^n \delta_{X_i}\delta_{Y_i}$. By definition, for any $f \in \mathcal{H}_X$ and $g \in \mathcal{H}_\mathcal{Y}$, the operator $\widehat{\Sigma}_{YX}^{(n)}$ gives the empirical covariance as follows;

$$\begin{aligned}
&\langle g, \widehat{\Sigma}_{YX}^{(n)} f\rangle_{\mathcal{H}_\mathcal{Y}} \\
&= \frac{1}{n}\sum_{i=1}^n \left\langle g, k_\mathcal{Y}(\cdot,Y_i) - \frac{1}{n}\sum_{s=1}^n k_\mathcal{Y}(\cdot,Y_s)\right\rangle_{\mathcal{H}_\mathcal{Y}} \left\langle k_X(\cdot,X_i) - \frac{1}{n}\sum_{t=1}^n k_X(\cdot,X_t), f\right\rangle_{\mathcal{H}_X} \\
&= \widehat{\mathrm{Cov}}[f(X), G(Y)].
\end{aligned}$$

Obviously, the rank of $\widehat{\Sigma}_{YX}^{(n)}$ is finite, because $\mathcal{R}(\widehat{\Sigma}_{YX}^{(n)})$ and $\mathcal{N}(\widehat{\Sigma}_{YX}^{(n)})^\perp$ are included in the linear hull of $\{k_\mathcal{Y}(\cdot,Y_i) - \frac{1}{n}\sum_{s=1}^n k_\mathcal{Y}(\cdot,Y_s)\}_{i=1}^n$ and $\{k_X(\cdot,X_i) - \frac{1}{n}\sum_{t=1}^n k_X(\cdot,X_t)\}_{i=1}^n$, respectively.

Let Q_X and Q_Y be the orthogonal projection which maps $\mathcal{H}_X$ onto $\overline{\mathcal{R}(\Sigma_{XX})}$ and $\mathcal{H}_\mathcal{Y}$ onto $\overline{\mathcal{R}(\Sigma_{YY})}$, respectively. It is known (Baker, 1973, Theorem 1) that Σ_{YX} has a representation

$$\Sigma_{YX} = \Sigma_{YY}^{1/2} V_{YX} \Sigma_{XX}^{1/2}, \tag{5}$$

where $V_{YX} : \mathcal{H}_X \to \mathcal{H}_\mathcal{Y}$ is a unique bounded operator such that $\|V_{YX}\| \le 1$ and $V_{YX} = Q_Y V_{YX} Q_X$. Note that the inverse of an operator may not exist in general, or may not be continuous if it exists. We often write V_{YX} by $\Sigma_{YY}^{-1/2}\Sigma_{YX}\Sigma_{XX}^{-1/2}$, however, by abuse of notation, even when $\Sigma_{XX}^{-1/2}$ or $\Sigma_{YY}^{-1/2}$ are not appropriately defined as operators.

2. Cross-covariance operator have been defined for Banach spaces by Baker (1973). However, we confine our discussion to RKHS.

2.3 Representation of Kernel CCA and Related Methods with Cross-covariance Operators

With cross-covariance operators for (X,Y), the kernel CCA problem can be formulated as

$$\sup_{f\in\mathcal{H}_X, g\in\mathcal{H}_Y} \langle g, \Sigma_{YX} f\rangle_{\mathcal{H}_Y} \quad \text{subject to} \quad \begin{cases} \langle f, \Sigma_{XX} f\rangle_{\mathcal{H}_X} = 1, \\ \langle g, \Sigma_{YY} g\rangle_{\mathcal{H}_Y} = 1. \end{cases}$$

As with classical CCA (Anderson, 2003, for example), the solution of the above kernel CCA problem is given by the eigenfunctions corresponding to the largest eigenvalue of the following generalized eigenproblem:

$$\begin{cases} \Sigma_{YX} f = \rho_1 \Sigma_{YY} g, \\ \Sigma_{XY} g = \rho_1 \Sigma_{XX} f. \end{cases} \tag{6}$$

For an i.i.d. sample $(X_1,Y_1),\ldots,(X_n,Y_n)$, the empirical estimator in Eq. (3) is

$$\sup_{f\in\mathcal{H}_X, g\in\mathcal{H}_Y} \langle g, \widehat{\Sigma}^{(n)}_{YX} f\rangle_{\mathcal{H}_Y} \quad \text{subject to} \quad \begin{cases} \langle f, (\widehat{\Sigma}^{(n)}_{XX} + \varepsilon_n I) f\rangle_{\mathcal{H}_X} = 1, \\ \langle g, (\widehat{\Sigma}^{(n)}_{YY} + \varepsilon_n I) g\rangle_{\mathcal{H}_Y} = 1, \end{cases}$$

and Eq. (6) becomes

$$\begin{cases} \widehat{\Sigma}^{(n)}_{YX} f = \widehat{\rho}^{(n)}_1 \big(\widehat{\Sigma}^{(n)}_{YY} + \varepsilon_n I\big) g, \\ \widehat{\Sigma}^{(n)}_{XY} g = \widehat{\rho}^{(n)}_1 \big(\widehat{\Sigma}^{(n)}_{XX} + \varepsilon_n I\big) f. \end{cases} \tag{7}$$

Let us assume that the operator V_{YX} given by Eq. (5) is compact,[3] and let ϕ and ψ be the unit eigenfunctions of V_{YX} corresponding to the largest singular value;[4] that is,

$$\langle \psi, V_{YX}\phi\rangle_{\mathcal{H}_Y} = \max_{\substack{f\in\mathcal{H}_X, g\in\mathcal{H}_Y \\ \|f\|_{\mathcal{H}_X} = \|g\|_{\mathcal{H}_Y} = 1}} \langle g, V_{YX} f\rangle_{\mathcal{H}_Y}. \tag{8}$$

Given that $\phi \in \mathcal{R}(\Sigma_{XX})$ and $\psi \in \mathcal{R}(\Sigma_{YY})$, it is easy to see from Eq. (6) that the solution of the kernel CCA is given by the inverse images[5]

$$f = \Sigma_{XX}^{-1/2}\phi, \qquad g = \Sigma_{YY}^{-1/2}\psi,$$

where f and g are determined up to an almost sure constant function. In the empirical case, let $\widehat{\phi}_n \in \mathcal{H}_X$ and $\widehat{\psi}_n \in \mathcal{H}_Y$ be the unit eigenfunctions corresponding to the largest singular value of the finite rank operator

$$\widehat{V}^{(n)}_{YX} := \big(\widehat{\Sigma}^{(n)}_{YY} + \varepsilon_n I\big)^{-1/2} \widehat{\Sigma}^{(n)}_{YX} \big(\widehat{\Sigma}^{(n)}_{XX} + \varepsilon_n I\big)^{-1/2}.$$

From Eq. (7), the empirical estimators $\widehat{f}_n$ and $\widehat{g}_n$ of kernel CCA are

$$\widehat{f}_n = (\widehat{\Sigma}^{(n)}_{XX} + \varepsilon_n I)^{-1/2}\widehat{\phi}_n, \qquad \widehat{g}_n = (\widehat{\Sigma}^{(n)}_{YY} + \varepsilon_n I)^{-1/2}\widehat{\psi}_n.$$

3. See Appendix A for compact operators.
4. While we presume that the eigenspaces are one dimensional in this section, we can easily relax it to multidimensional spaces by considering the eigenspaces corresponding to the largest eigenvalues. See the remarks after Theorem 2.
5. The operators $\Sigma_{XX}^{1/2}$ and $\Sigma_{YY}^{1/2}$ may not be invertible, but their inverses are well-defined up to an almost sure constant function when applied to functions belonging to the respective ranges of $\Sigma_{XX}^{1/2}$ and $\Sigma_{YY}^{1/2}$.

The empirical operators and the estimators described above can be expressed using *Gram matrices*, as is often done in kernel methods. The solutions $\widehat{f}_n$ and $\widehat{g}_n$ are exactly the same as those given in Bach and Jordan (2002), as we confirm below. Let $u_i \in \mathcal{H}_\mathcal{X}$ and $v_i \in \mathcal{H}_\mathcal{Y}$ $(1 \le i \le n)$ be functions defined by

$$u_i = k_\mathcal{X}(\cdot, X_i) - \frac{1}{n}\sum_{j=1}^n k_\mathcal{X}(\cdot, X_j), \qquad v_i = k_\mathcal{Y}(\cdot, Y_i) - \frac{1}{n}\sum_{j=1}^n k_\mathcal{Y}(\cdot, Y_j).$$

Because $\mathcal{R}(\widehat{\Sigma}_{XX}^{(n)})$ and $\mathcal{R}(\widehat{\Sigma}_{YY}^{(n)})$ are spanned by $(u_i)_{i=1}^n$ and $(v_i)_{i=1}^n$, respectively, the eigenfunctions of $\widehat{V}_{YX}^{(n)}$ are given by a linear combination of u_i and v_i. Letting $\phi = \sum_{i=1}^n \alpha_i u_i$ and $\psi = \sum_{i=1}^n \beta_i v_i$, direct calculation of $\langle \psi, \widehat{V}_{YX}^{(n)} \phi \rangle_{\mathcal{H}_\mathcal{Y}}$ shows that the eigenfunctions $\widehat{\phi}_n$ and $\widehat{\psi}_n$ of $\widehat{\Sigma}_{YX}^{(n)}$ corresponding to the largest singular value are given by the coefficients $\widehat{\alpha}$ and $\widehat{\beta}$ that satisfy

$$\max_{\substack{\alpha,\beta \in \mathbb{R}^n \\ \alpha^T G_X \alpha = \beta^T G_Y \beta = 1}} \beta^T \big(G_Y + n\varepsilon_n I_n\big)^{-1/2} G_Y G_X \big(G_X + n\varepsilon_n I_n\big)^{-1/2} \alpha,$$

where G_X is the centered Gram matrix,

$$(G_X)_{ij} = k_\mathcal{X}(X_i, X_j) - \frac{1}{n}\sum_{b=1}^n k_\mathcal{X}(X_i, X_b) - \frac{1}{n}\sum_{a=1}^n k_\mathcal{X}(X_a, X_j) + \frac{1}{n^2}\sum_{a=1}^n\sum_{b=1}^n k_\mathcal{X}(X_a, X_b),$$

with G_Y defined accordingly. The solution of the kernel CCA problem is

$$\widehat{f}_n = (\widehat{\Sigma}_{XX}^{(n)} + \varepsilon_n I)^{-1/2} \widehat{\phi}_n = \sum_{i=1}^n \widehat{\xi}_i u_i, \qquad \widehat{g}_n = (\widehat{\Sigma}_{YY}^{(n)} + \varepsilon_n I)^{-1/2} \widehat{\psi}_n = \sum_{i=1}^n \widehat{\zeta}_i v_i,$$

where

$$\widehat{\xi} = \sqrt{n}(G_X + n\varepsilon_n I_n)^{-1/2}\widehat{\alpha} \quad \text{and} \quad \widehat{\zeta} = \sqrt{n}(G_Y + n\varepsilon_n I_n)^{-1/2}\widehat{\beta}.$$

Thus, the linear coefficients $\widehat{\xi}$ and $\widehat{\zeta}$ are the solution of

$$\max_{\substack{\xi,\zeta \in \mathbb{R}^n \\ \xi^T (G_X^2 + n\varepsilon_n G_X)\xi = \zeta^T (G_Y^2 + n\varepsilon_n G_Y)\zeta = n}} \zeta^T G_Y G_X \xi,$$

which is exactly the same as the one proposed by Bach and Jordan (2002). Bach and Jordan approximate $(G_X^2 + n\varepsilon_n G_X)$ by $(G_X + \frac{n\varepsilon_n}{2} I_n)^2$ for computational simplicity. Note that our theoretical results in the next section still hold with this approximation, because this modification causes only higher order changes in $\widehat{\alpha}$ and $\widehat{\beta}$, which perturbs the empirical eigenfunctions $\widehat{\phi}_n$, $\widehat{\psi}_n$, $\widehat{f}_n$, and $\widehat{g}_n$ only in higher order.

There are additional, related methods to extract nonlinear dependence of two random variables with positive definite kernels. The Constrained Covariance (COCO, Gretton et al., 2005b) uses the unit eigenfunctions of the cross-covariance operator Σ_{YX}. Thus the solution of COCO is

$$\max_{\substack{f \in \mathcal{H}_\mathcal{X}, g \in \mathcal{H}_\mathcal{Y} \\ \|f\|_{\mathcal{H}_\mathcal{X}} = \|g\|_{\mathcal{H}_\mathcal{Y}} = 1}} \langle g, \Sigma_{YX} f \rangle_{\mathcal{H}_\mathcal{Y}} = \max_{\substack{f \in \mathcal{H}_\mathcal{X}, g \in \mathcal{H}_\mathcal{Y} \\ \|f\|_{\mathcal{H}_\mathcal{X}} = \|g\|_{\mathcal{H}_\mathcal{Y}} = 1}} \mathrm{Cov}[f(X), g(Y)].$$

The consistency of COCO has been proved by Gretton et al. (2005a). Unlike kernel CCA, COCO normalizes the covariance by the RKHS norms of f and g. Kernel CCA is a more direct nonlinear extension of the ordinary CCA than COCO. COCO tends to find functions with large variance for $f(X)$ and $g(Y)$, which may not be the most correlated features. On the other hand, kernel CCA may encounter situations where it finds functions with moderately large covariance but very small variances for $f(X)$ or $g(Y)$, since Σ_{XX} and Σ_{YY} can have arbitrarily small eigenvalues.

A possible compromise between these methods is to use ϕ and ψ in Eq. (8), and their estimates $\widehat{\phi}_n$ and $\widehat{\psi}_n$. While the statistical meaning of this approach is not as direct as kernel CCA, it can incorporate the normalization by Σ_{XX} and Σ_{YY}. We call this variant *NOrmalized Cross-Covariance Operator* (NOCCO). We will establish the consistency of kernel CCA and NOCCO in the next section, and give experimental comparisons of these methods in Section 4.

3. Main Theorems

First, the following theorem asserts the consistency of the estimator of NOCCO in the RKHS norm, when the regularization parameter ε_n goes to zero slowly enough.

Theorem 1 *Let $(\varepsilon_n)_{n=1}^{\infty}$ be a sequence of positive numbers such that*

$$\lim_{n\to\infty} \varepsilon_n = 0, \qquad \lim_{n\to\infty} \frac{n^{-1/3}}{\varepsilon_n} = 0. \tag{9}$$

Assume V_{YX} is a compact operator and the eigenspaces which attain the singular value problem

$$\max_{\substack{\phi\in\mathcal{H}_X,\psi\in\mathcal{H}_Y \\ \|\phi\|_{\mathcal{H}_X}=\|\psi\|_{\mathcal{H}_Y}=1}} \langle \psi, V_{YX}\phi\rangle_{\mathcal{H}_Y}$$

are one-dimensional. Let $\widehat{\phi}_n$ and $\widehat{\psi}_n$ be the unit eigenfunctions for the largest singular value of $\widehat{V}_{YX}^{(n)}$. Then,

$$|\langle \widehat{\phi}_n, \phi\rangle_{\mathcal{H}_X}| \to 1, \qquad |\langle \widehat{\psi}_n, \psi\rangle_{\mathcal{H}_Y}| \to 1$$

in probability, as n goes to infinity.

The next main result shows the convergence of kernel CCA in the norm of $L_2(P_X)$ and $L_2(P_Y)$.

Theorem 2 *Let $(\varepsilon_n)_{n=1}^{\infty}$ be a sequence of positive numbers which satisfies Eq. (9). Assume that ϕ and ψ are included in $\mathcal{R}(\Sigma_{XX})$ and $\mathcal{R}(\Sigma_{YY})$, respectively, and that V_{YX} is compact. Then,*

$$\left\|(\widehat{f}_n - E_X[\widehat{f}_n(X)]) - (f - E_X[f(X)])\right\|_{L_2(P_X)} \to 0$$

and

$$\left\|(\widehat{g}_n - E_Y[\widehat{g}_n(Y)]) - (g - E_Y[g(Y)])\right\|_{L_2(P_Y)} \to 0$$

in probability, as n goes to infinity.

While in the above theorems we confine our attention to the first eigenfunctions, it is not difficult to verify the convergence of eigenspaces corresponding to the m-th largest eigenvalue by extending Lemma 10 in the Appendix. See also the remark after the lemma.

The convergence of NOCCO in RKHS norm is a very strong result. If $\mathcal{X}$ and $\mathcal{Y}$ are topological spaces, and if the kernels $k_{\mathcal{X}}$ and $k_{\mathcal{Y}}$ are continuous and bounded, all the functions in $\mathcal{H}_{\mathcal{X}}$ and $\mathcal{H}_{\mathcal{Y}}$ are continuous and the RKHS norm is stronger than the uniform norm in $C(\mathcal{X})$ and $C(\mathcal{Y})$, where $C(\mathcal{Z})$ is the Banach space of all the continuous functions on a topological space $\mathcal{Z}$ with the supremum norm. In fact, for any $f \in \mathcal{H}_{\mathcal{X}}$, we have $\sup_{x\in\mathcal{X}} |f(x)| = \sup_{x\in\mathcal{X}} |\langle k_{\mathcal{X}}(\cdot,x), f\rangle_{\mathcal{H}_{\mathcal{X}}}| \leq \sup_{x\in\mathcal{X}}(k_{\mathcal{X}}(x,x))^{1/2}\|f\|_{\mathcal{H}_{\mathcal{X}}}$. In such cases, Theorem 1 implies $\widehat{\phi}_n$ and $\widehat{\psi}_n$ converge uniformly to ϕ and ψ, respectively. This uniform convergence is useful in practice, because in many applications the function value at each point is important.

The above theorems assume the compactness of V_{YX}, which requires that for any complete orthonormal systems (CONS) $\{\phi_i\}_{i=1}^{\infty}$ of $\mathcal{H}_{\mathcal{X}}$ and $\{\psi_i\}_{i=1}^{\infty}$ of $\mathcal{H}_{\mathcal{Y}}$, the correlation of $\Sigma_{XX}^{-1/2}\phi_i(X)$ and $\Sigma_{YY}^{-1/2}\psi_i(Y)$ decay to zero as $i \to \infty$. This is not necessarily satisfied in general. A trivial example is the case of variables with $Y = X$. In this case, $V_{YX} = I$ is not compact, and the problem in Theorem 1 is solved by an arbitrary function. In this situation, the kernel CCA problem in Theorem 2 does not have solutions if Σ_{XX} has arbitrarily small eigenvalues.

We give a useful sufficient condition that V_{YX} is Hilbert-Schmidt, which necessarily implies compactness. The condition is described in terms of mean square contingency, which is one of the standard criteria to measure the dependency of two random variables (Rényi, 1970). It is known (Buja, 1990) that the covariance operator considered on L^2 is Hilbert-Schmidt if the mean square contingency is finite. We modify the result to the case of the covariance operator on RKHS.

Assume that the measure spaces $(\mathcal{X}, \mathcal{B}_{\mathcal{X}})$ and $(\mathcal{Y}, \mathcal{B}_{\mathcal{Y}})$ admit measures $\mu_{\mathcal{X}}$ and $\mu_{\mathcal{Y}}$, respectively, and that P_{XY} is absolutely continuous with respect to the product measure $\mu_{\mathcal{X}} \times \mu_{\mathcal{Y}}$ with a probability density function $p_{XY}(x,y)$. Let $\zeta(x,y)$ be a function on $\mathcal{X} \times \mathcal{Y}$ defined by

$$\zeta(x,y) = \frac{p_{XY}(x,y)}{p_X(x)p_Y(y)} - 1,$$

where $p_X(x)$ and $p_Y(y)$ are the probability density functions of the marginal distributions P_X and P_Y, respectively. The *mean square contingency* $C(X,Y)$ is defined by

$$C(X,Y) = \left\{\int\int \zeta(x,y)^2 dP_X dP_Y\right\}^{1/2}.$$

It is easy to see $C(X,Y) = 0$ if and only if X and Y are independent. Obviously we have

$$C(X,Y)^2 = \int\int \frac{p_{XY}(x,y)^2}{p_X(x)p_Y(y)} d\mu_{\mathcal{X}} d\mu_{\mathcal{Y}} - 1 = \int \zeta(x,y) dP_{XY}.$$

Thus, $C(X,Y)^2$ is an upper bound of the mutual information $MI(X,Y) = \int \log \frac{p_{XY}(x,y)}{p_X(x)p_Y(y)} dP_{XY}$, because $\log(z+1) \leq z$ for $z > 0$.

Theorem 3 *Suppose that the measurable spaces $(\mathcal{X}, \mathcal{B}_{\mathcal{X}})$ and $(\mathcal{Y}, \mathcal{B}_{\mathcal{Y}})$ have measures $\mu_{\mathcal{X}}$ and $\mu_{\mathcal{Y}}$, respectively, so that P_{XY} is absolutely continuous with respect to the product measure $\mu_{\mathcal{X}} \times \mu_{\mathcal{Y}}$ with a probability density function $p_{XY}(x,y)$. If the mean square contingency $C(X,Y)$ is finite, that is, if*

$$\int\int \frac{p_{XY}(x,y)^2}{p_X(x)p_Y(y)} d\mu_{\mathcal{X}} d\mu_{\mathcal{Y}} < \infty,$$

then the operator $V_{YX} : \mathcal{H}_{\mathcal{X}} \to \mathcal{H}_{\mathcal{Y}}$ is Hilbert-Schmidt, and

$$\|V_{YX}\|_{HS} \leq C(X,Y) = \|\zeta\|_{L^2(P_X \times P_Y)}.$$

The proof is given in Section 5. The assumption that the mean square contingency is finite is very natural when we consider the dependence of two different random variables, as in the situation where kernel CCA is applied. It is interesting to see that Breiman and Friedman (1985) also discuss a similar condition for the existence of optimal functions for functional canonical correlation.

He et al. (2003) discuss the eigendecomposition of the operator V_{YX}. They regard random processes in L^2 spaces as data. In our case, transforms of the original random variables, $k_{\mathcal{X}}(\cdot, X)$ and $k_{\mathcal{Y}}(\cdot, Y)$, also induce processes in RKHS. He et al. (2003) give a condition for the eigendecomposition of V_{YX} in terms of the eigendecomposition of the data processes, while our condition is a more direct property of the original random variables.

Leurgans et al. (1993) discuss canonical correlation analysis on curves, which are represented by stochastic processes on an interval, and use the Sobolev space of functions with square integrable second derivative. Since this Sobolev space is an RKHS, their method is an example of kernel CCA in a specific RKHS. They also prove the consistency of estimators under the condition $n^{-1/2}/\varepsilon_n \to 0$. Although the proof can be extended to a general RKHS, the convergence is measured by that of the correlation,

$$\frac{\left|\langle \widehat{f}_n, \Sigma_{XX} f\rangle_{\mathcal{H}_X}\right|}{\left(\langle \widehat{f}_n, \Sigma_{XX}\widehat{f}_n\rangle_{\mathcal{H}_X}\right)^{1/2}\left(\langle f, \Sigma_{XX} f\rangle_{\mathcal{H}_X}\right)^{1/2}} \quad \to \quad 1.$$

Note that in the denominator the population covariance $\langle \widehat{f}_n, \Sigma_{XX}\widehat{f}_n\rangle_{\mathcal{H}_X}$ is used, which is not computable in practice. The above convergence of correlation is weaker than the L_2 convergence in Theorem 2. In fact, since the desired eigenfunction f is normalized so that $\langle f, \Sigma_{XX} f\rangle_{\mathcal{H}_X} = 1$, it is easy to derive the above convergence of correlation from Theorem 2. On the other hand, the convergence of correlation does not imply $\langle (\widehat{f}_n - f), \Sigma_{XX}(\widehat{f}_n - f)\rangle_{\mathcal{H}_X}$. From the equality

$$\begin{aligned}\langle (\widehat{f}_n - f), \Sigma_{XX}(\widehat{f}_n - f)\rangle_{\mathcal{H}_X} &= \left(\langle \widehat{f}_n, \Sigma_{XX}\widehat{f}_n\rangle_{\mathcal{H}_X}^{1/2} - \langle f, \Sigma_{XX} f\rangle_{\mathcal{H}_X}^{1/2}\right)^2 \\ &\quad + 2\left(1 - \frac{\langle \widehat{f}_n, \Sigma_{XX} f\rangle_{\mathcal{H}_X}}{\|\Sigma_{XX}^{1/2}\widehat{f}_n\|_{\mathcal{H}_X}\|\Sigma_{XX}^{1/2} f\|_{\mathcal{H}_X}}\right)\|\Sigma_{XX}^{1/2}\widehat{f}_n\|_{\mathcal{H}_X}\|\Sigma_{XX}^{1/2} f\|_{\mathcal{H}_X},\end{aligned}$$

we require the convergence $\langle \widehat{f}_n, \Sigma_{XX}\widehat{f}_n\rangle_{\mathcal{H}_X} \to \langle f, \Sigma_{XX} f\rangle_{\mathcal{H}_X} = 1$ in order to guarantee the left hand side converges to zero. With the normalization $\langle \widehat{f}_n, (\widehat{\Sigma}_{XX}^{(n)} + \varepsilon_n I)\widehat{f}_n\rangle_{\mathcal{H}_X} = \langle f, \Sigma_{XX} f\rangle_{\mathcal{H}_X} = 1$, however, the convergence of $\langle \widehat{f}_n, \Sigma_{XX}\widehat{f}_n\rangle_{\mathcal{H}_X}$ is not clear. We use the stronger assumption $n^{-1/3}/\varepsilon_n \to 0$ to prove $\langle (\widehat{f}_n - f), \Sigma_{XX}(\widehat{f}_n - f)\rangle_{\mathcal{H}_X} \to 0$ in Theorem 2.

4. Numerical Simulations

In this section, we show results of numerical simulations for kernel CCA and related methods. We use a synthetic data set for which the optimal nonlinear functions in the population kernel CCA (Eq. (2)) are explicitly known, and demonstrate the convergence behavior for various values of ε_n. For the quantitative evaluation of convergence, we consider only kernel CCA, because the exact solutions for NOCCO or COCO in population are not known in closed form.

To generate our test data, we provide two univariate random variables X and Y for which the true transforms $f(X)$ and $g(Y)$ are highly linearly correlated for some $f \in \mathcal{H}_X$ and $g \in \mathcal{H}_{\mathcal{Y}}$. We generate a sample from P_{XY} as follows: first, we sample $Z_1, \ldots, Z_n$ uniformly on the unit interval $[0,1]$. Next, we derive two i.i.d. linearly correlated random samples U_i and V_i from these Z_i. Finally,

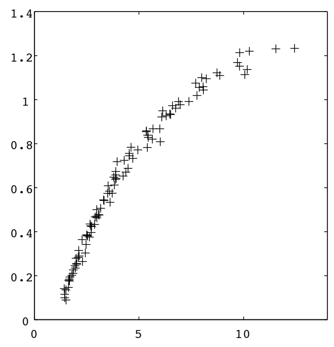


Figure 2: The plot of (R_i^X, R_i^Y) of the data used in the experiment.

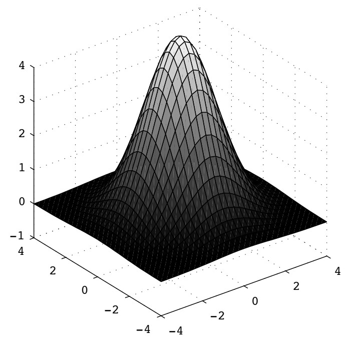
we transform these variables to radius data R_i^X and R_i^Y by the inverse of the Gaussian function $\exp(-aR^2)$ for some $a > 0$. The explicit form of these relations are

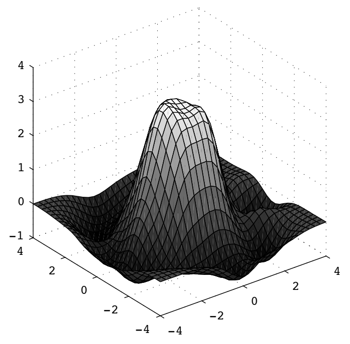
$$U_i = Z_i + 0.06 + e_i^X, \quad V_i = Z_i + 3 + e_i^Y,$$
$$R_i^X = \left(-4\log(U_i/1.5)\right)^{1/2}, \qquad R_i^Y = \left(-4\log(V_i/4.1)\right)^{1/2},$$

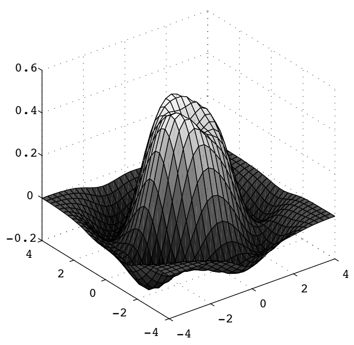
where e_i^X and e_i^Y are independent noise following a zero-mean saturated Gaussian distribution so that U_i and V_i are positive. See Figure 2 for an example data set. The samples X_i and Y_i are taken uniformly on the 2 dimensional circles with the radius R_i^X and R_i^Y, respectively. Thus, the maximum canonical correlation in population is attained by $f(x) = 1.5\exp(-\frac{1}{4}\|x\|^2)$ and $g(y) = 4.1\exp(-\frac{1}{4}\|y\|^2)$ up to scale and shift.

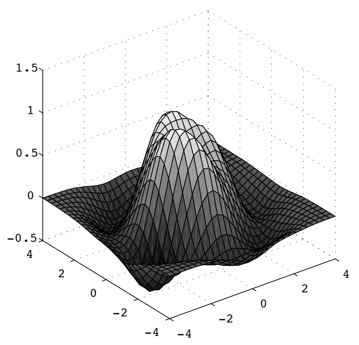
We perform kernel CCA, NOCCO, and COCO with Gaussian RBF kernel $k(x,y) = \exp(-\|x - y\|^2)$ on the data. Note that the true functions f and g for kernel CCA are included in RKHS with this kernel. The graphs of resulting functions for X, the true function $f(x)$, and the transformed data are shown in Figure 3. We see that the functions obtained by Kernel CCA, NOCCO, and COCO have a similar shape to $f(x)$. Note that, because the data exist only around the area $f(x) \leq 1.0$, the estimation accuracy in the flat area including the origin is low. In the plots (e)-(g), kernel CCA gives linearly correlated feature vectors, while NOCCO and COCO do not aim at obtaining linearly correlated vectors. However, we see that these two methods also give the features that contain the sufficient information on the dependency between X and Y.

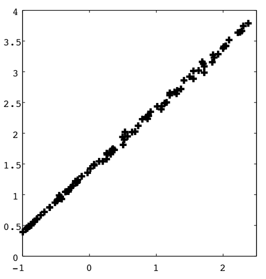
Next, we conduct numerical simulations to verify the convergence rate of kernel CCA. Figure 4 shows the convergence to the true functions for various decay rates of the regularization coefficient $\varepsilon_n = 0.001 \times n^{-a}$ with $a = 0.1 \sim 0.8$. For the estimated functions $\widehat{f}_n$ and $\widehat{g}_n$ with the data sizes $n = 10$, 25, 50, 75, 100, 250, 500, 750, and 1000, the $L^2(P_X)$ and $L^2(P_{\mathcal{Y}})$ distances between the estimated and true functions are evaluated by generating 10000 samples from the true distribution. The curves show an average over 30 experiments with different random data. It should be noted that, although the theoretical sufficient condition for convergence requires a slower order of ε_n than $n^{-1/3}$, faster orders give better convergence in these simulations. The convergence is best at $a = 0.6$, and becomes worse for faster decay rates; the optimum rate likely depends on the statistical properties of the data. It might therefore be interesting to find the best rate or best value of ε_n for the given data, although this is beyond the scope of the present paper.

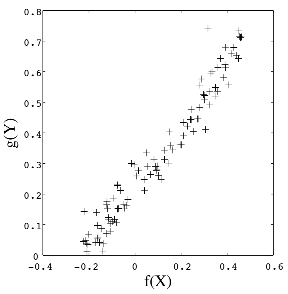

(a) True function $f(x)$


(b) KCCA

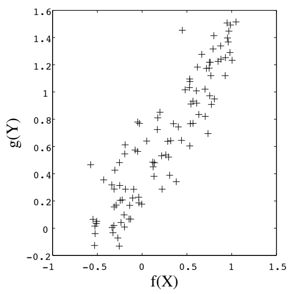

(c) NOCCO


(d) COCO


(e) KCCA



(f) NOCCO



(g) COCO

Figure 3: The true function $f(x)$ and the estimated functions based on 100 data points are shown in (a)-(d). The plots of the transformed data $(\widehat{f}_n(X_i), \widehat{g}_n(Y_i))$ are given in (e)-(g). Note that in (e)-(g) the clear linear correlation is seen in (e), only because it is the criterion of the kernel CCA; the other two methods use different criterion, but still show strong correlation.

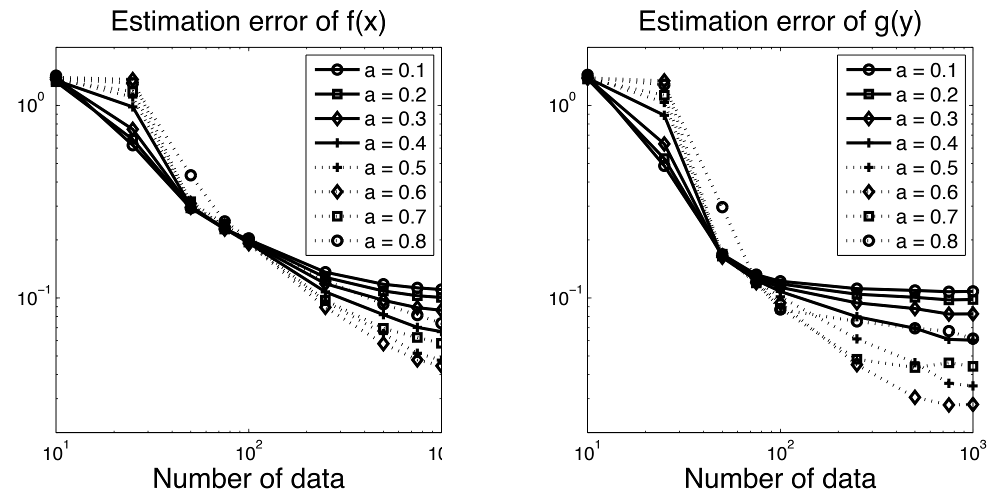


Figure 4: L^2 distances between the true function and its estimate using kernel CCA. The regularization parameter is $\varepsilon_n = 0.001 \times n^{-a}$, where the decay rate a ranges from 0.1 to 0.8.

5. Proof of the Main Theorems

In this section, we prove the main theorems in Section 3.

5.1 Hilbert-Schmidt Norm of Covariance Operators

Preliminary to the proofs, in this subsection we show some results on the Hilbert-Schmidt norm of cross-covariance operators. For convenience, we provide the definition and some basic properties of Hilbert-Schmidt operators in the Appendix. See also Gretton et al. (2005a).

We begin with a brief introduction to random elements in a Hilbert space (Vakhania et al., 1987; Baker, 1973). Let $\mathcal{H}$ be a Hilbert space equipped with Borel σ-field. A *random element* in the Hilbert space $\mathcal{H}$ is a measurable map $F : \Omega \to \mathcal{H}$ from a measurable space $(\Omega, \mathfrak{S})$. Let $\mathcal{H}$ be an RKHS on a measurable set $\mathcal{X}$ with a measurable positive definite kernel k. For a random variable X in $\mathcal{X}$, the map $k(\cdot, X)$ defines a random element in $\mathcal{H}$.

A random element F in a Hilbert space $\mathcal{H}$ is said to have *strong order* p ($0 < p < \infty$) if $E\|F\|^p$ is finite. For a random element F of strong order one, the expectation of F is defined as the element m_F in $\mathcal{H}$ such that

$$\langle m_F, g \rangle_{\mathcal{H}} = E[\langle F, g \rangle_{\mathcal{H}}]$$

holds for all $g \in \mathcal{H}$. The existence and the uniqueness of the mean element is a consequence of Riesz's representation theorem. The expectation m_F is denoted by $E[F]$. Then, the equality $\langle E[F], g \rangle_{\mathcal{H}} = E[\langle F, g \rangle_{\mathcal{H}}]$ is justified, which means the expectation and the inner product are interchangeable. If F and G have strong order two, $E[|\langle F, G \rangle_{\mathcal{H}}|]$ is finite. If further F and G are independent, the relation

$$E[\langle F, G \rangle_{\mathcal{H}}] = \langle E[F], E[G] \rangle_{\mathcal{H}} \tag{10}$$

holds.

It is easy to see that the example $F = k(\cdot,X)$ in an RKHS $\mathcal{H}$ has strong order two, that is, $E[\|F\|^2] < \infty$, under the assumption $E[k(X,X)] < \infty$. The expectation of $k(\cdot,X)$ is equal to m_X in Eq. (4) by definition. For two RKHS $\mathcal{H}_\mathcal{X}$ on $\mathcal{X}$ and $\mathcal{H}_\mathcal{Y}$ on $\mathcal{Y}$ with kernels $k_\mathcal{X}$ and $k_\mathcal{Y}$, respectively, under the conditions Eq. (1), the random element $k_\mathcal{X}(\cdot,X)k_\mathcal{Y}(\cdot,Y)$ in the direct product $\mathcal{H}_\mathcal{X} \otimes \mathcal{H}_\mathcal{Y}$ has strong order one.

The following lemma is straightforward from Lemma 1 in Gretton et al. (2005a) and Eq. (10). See Appendix for definitions of the Hilbert-Schmidt operator and Hilbert-Schmidt norm.

Lemma 4 *The cross-covariance operator Σ_{YX} is a Hilbert-Schmidt operator, and its Hilbert-Schmidt norm is given by*

$$\begin{aligned}
&\|\Sigma_{YX}\|_{HS}^2 \\
&= E_{YX}E_{\tilde{Y}\tilde{X}}\left[\langle k_\mathcal{X}(\cdot,X) - m_X, k_\mathcal{X}(\cdot,\tilde{X}) - m_X\rangle_{\mathcal{H}_\mathcal{X}}\langle k_\mathcal{Y}(\cdot,\tilde{Y}) - m_Y, k_\mathcal{Y}(\cdot,Y) - m_Y\rangle_{\mathcal{H}_\mathcal{Y}}\right] \\
&= \left\|E_{YX}[(k_\mathcal{X}(\cdot,X) - m_X)(k_\mathcal{Y}(\cdot,Y) - m_Y)]\right\|_{\mathcal{H}_\mathcal{X}\otimes\mathcal{H}_\mathcal{Y}}^2
\end{aligned}$$

where $(\tilde{X},\tilde{Y})$ and (X,Y) are independently and identically distributed with distribution P_{XY}.

From the facts $\mathcal{H}_\mathcal{X} \subset L_2(P_X)$ and $\mathcal{H}_\mathcal{Y} \subset L_2(P_Y)$, the law of large numbers implies for each $f \in \mathcal{H}_\mathcal{X}$ and $g \in \mathcal{H}_\mathcal{Y}$

$$\lim_{n\to\infty}\langle g, \widehat{\Sigma}_{YX}^{(n)} f\rangle_{\mathcal{H}_\mathcal{Y}} = \langle g, \Sigma_{YX} f\rangle_{\mathcal{H}_\mathcal{Y}}$$

in probability. Moreover, the central limit theorem shows the above convergence is of order[6] $O_p(n^{-1/2})$. The following lemma shows the tight uniform result that $\|\widehat{\Sigma}_{YX}^{(n)} - \Sigma_{YX}\|_{HS}$ converges to zero in the order of $O_p(n^{-1/2})$.

Lemma 5

$$\left\|\widehat{\Sigma}_{YX}^{(n)} - \Sigma_{YX}\right\|_{HS} = O_p(n^{-1/2}) \quad (n \to \infty).$$

Proof Write for simplicity $F = k_\mathcal{X}(\cdot,X) - E_X[k_\mathcal{X}(\cdot,X)]$, $G = k_\mathcal{Y}(\cdot,Y) - E_Y[k_\mathcal{Y}(\cdot,Y)]$, $F_i = k_\mathcal{X}(\cdot,X_i) - E_X[k_\mathcal{X}(\cdot,X)]$, $G_i = k_\mathcal{Y}(\cdot,Y_i) - E_Y[k_\mathcal{Y}(\cdot,Y)]$, and $\mathcal{F} = \mathcal{H}_\mathcal{X} \otimes \mathcal{H}_\mathcal{Y}$. Then, $F, F_1, \ldots, F_n$ are i.i.d. random elements in $\mathcal{H}_\mathcal{X}$, and a similar fact holds for $G, G_1, \ldots, G_n$. Lemma 4 implies

$$\|\widehat{\Sigma}_{YX}^{(n)}\|_{HS}^2 = \left\|\frac{1}{n}\sum_{i=1}^n\Big(F_i - \frac{1}{n}\sum_{j=1}^n F_j\Big)\Big(G_i - \frac{1}{n}\sum_{j=1}^n G_j\Big)\right\|_\mathcal{F}^2,$$

and the same argument as in the proof of the lemma yields

$$\langle\Sigma_{YX}, \widehat{\Sigma}_{YX}^{(n)}\rangle_{HS} = \Big\langle E[FG], \frac{1}{n}\sum_{i=1}^n\Big(F_i - \frac{1}{n}\sum_{j=1}^n F_j\Big)\Big(G_i - \frac{1}{n}\sum_{j=1}^n G_j\Big)\Big\rangle_\mathcal{F}.$$

From these equations, we have

$$\begin{aligned}
\left\|\widehat{\Sigma}_{YX}^{(n)} - \Sigma_{YX}\right\|_{HS}^2 &= \left\|\Sigma_{YX}\right\|_{HS}^2 - 2\langle\Sigma_{YX}, \widehat{\Sigma}_{YX}^{(n)}\rangle_{HS} + \left\|\widehat{\Sigma}_{YX}^{(n)}\right\|_{HS}^2 \\
&= \left\|\frac{1}{n}\sum_{i=1}^n\Big(F_i - \frac{1}{n}\sum_{j=1}^n F_j\Big)\Big(G_i - \frac{1}{n}\sum_{j=1}^n G_j\Big) - E[FG]\right\|_\mathcal{F}^2 \\
&= \left\|\frac{1}{n}\sum_{i=1}^n F_iG_i - E[FG] - \Big(2 - \frac{1}{n}\Big)\Big(\frac{1}{n}\sum_{i=1}^n F_i\Big)\Big(\frac{1}{n}\sum_{i=1}^n G_i\Big)\right\|_\mathcal{F}^2,
\end{aligned}$$

6. A random variable Z_n is said to be of order $O_p(a_n)$ if for any $\varepsilon > 0$ there exists $M > 0$ such that $\sup_n \Pr(|Z_n| > Ma_n) < \varepsilon$. See, for example, van der Vaart (1998).

which provides a bound

$$\big\|\widehat{\Sigma}_{YX}^{(n)}-\Sigma_{YX}\big\|_{HS}\le\Big\|\frac{1}{n}\sum_{i=1}^{n}F_iG_i-E[FG]\Big\|_{\mathcal{F}}+2\Big\|\Big(\frac{1}{n}\sum_{i=1}^{n}F_i\Big)\Big(\frac{1}{n}\sum_{i=1}^{n}G_i\Big)\Big\|_{\mathcal{F}}. \tag{11}$$

Let $Z_i=F_iG_i-E[FG]$. Since the variance of a sum of independent random variables is equal to the sum of their variances, we obtain

$$E\Big\|\frac{1}{n}\sum_{i=1}^{n}Z_i\Big\|_{\mathcal{F}}^2=\frac{1}{n}E\|Z_1\|_{\mathcal{F}}^2, \tag{12}$$

which is of order $O(1/n)$ because $E\|Z_1\|_{\mathcal{F}}^2<\infty$. From the inequality

$$\begin{aligned}E\Big\|\Big(\frac{1}{n}\sum_{i=1}^{n}F_i\Big)\Big(\frac{1}{n}\sum_{i=1}^{n}G_i\Big)\Big\|_{\mathcal{F}}&=E\Big[\Big\|\frac{1}{n}\sum_{i=1}^{n}F_i\Big\|_{\mathcal{H}_X}\Big\|\frac{1}{n}\sum_{i=1}^{n}G_i\Big\|_{\mathcal{H}_Y}\Big]\\&\le\Big(E\Big\|\frac{1}{n}\sum_{i=1}^{n}F_i\Big\|_{\mathcal{H}_X}^2\Big)^{1/2}\Big(E\Big\|\frac{1}{n}\sum_{i=1}^{n}G_i\Big\|_{\mathcal{H}_Y}^2\Big)^{1/2},\end{aligned}$$

in a similar way to Eq. (12), we have $E\big\|\big(\frac{1}{n}\sum_{i=1}^{n}F_i\big)\big(\frac{1}{n}\sum_{i=1}^{n}G_i\big)\big\|_{\mathcal{F}}=O(1/n)$.

From Eq. (11), we have $E\big\|\widehat{\Sigma}_{YX}^{(n)}-\Sigma_{YX}\big\|_{HS}=O(1/\sqrt{n})$, and the proof is completed by Chebyshev's inequality. ■

5.2 Preliminary Lemmas

For the proof of the main theorems, we show the empirical estimate $\widehat{V}_{YX}^{(n)}$ converges in norm to the normalized cross-covariance operator $V_{YX}=\Sigma_{YY}^{-1/2}\Sigma_{YX}\Sigma_{XX}^{-1/2}$ for an appropriate order of the regularization coefficient ε_n. We divide the task into two lemmas: the first evaluates the difference between the empirical estimate $\widehat{V}_{YX}^{(n)}$ and a regularized version of V_{YX}, and the second asserts that the regularized version converges to V_{YX} if ε_n goes to zero at the appropriate rate.

Lemma 6 *Let ε_n be a positive number such that $\varepsilon_n\to 0$ ($n\to\infty$). Then, for the i.i.d. sample $(X_1,Y_1),\dots,(X_n,Y_n)$, we have*

$$\big\|\widehat{V}_{YX}^{(n)}-(\Sigma_{YY}+\varepsilon_nI)^{-1/2}\Sigma_{YX}(\Sigma_{XX}+\varepsilon_nI)^{-1/2}\big\|=O_p(\varepsilon_n^{-3/2}n^{-1/2}).$$

Proof The operator in the left hand side is decomposed as

$$\begin{aligned}&\widehat{V}_{YX}^{(n)}-(\Sigma_{YY}+\varepsilon_nI)^{-1/2}\Sigma_{YX}(\Sigma_{XX}+\varepsilon_nI)^{-1/2}\\&=\big\{(\widehat{\Sigma}_{YY}^{(n)}+\varepsilon_nI)^{-1/2}-(\Sigma_{YY}+\varepsilon_nI)^{-1/2}\big\}\widehat{\Sigma}_{YX}^{(n)}(\widehat{\Sigma}_{XX}^{(n)}+\varepsilon_nI)^{-1/2}\\&\quad+(\Sigma_{YY}+\varepsilon_nI)^{-1/2}\big\{\widehat{\Sigma}_{YX}^{(n)}-\Sigma_{YX}\big\}(\widehat{\Sigma}_{XX}^{(n)}+\varepsilon_nI)^{-1/2}\\&\quad+(\Sigma_{YY}+\varepsilon_nI)^{-1/2}\Sigma_{YX}\big\{(\widehat{\Sigma}_{XX}^{(n)}+\varepsilon_nI)^{-1/2}-(\Sigma_{XX}+\varepsilon_nI)^{-1/2}\big\}.\end{aligned} \tag{13}$$

From the equality

$$A^{-1/2}-B^{-1/2}=A^{-1/2}\big(B^{3/2}-A^{3/2}\big)B^{-3/2}+(A-B)B^{-3/2},$$

the first term in the right hand side of Eq. (13) is equal to

$$\Big\{(\widehat{\Sigma}_{YY}^{(n)}+\varepsilon_n I)^{-1/2}\big\{\big(\Sigma_{YY}+\varepsilon_n I\big)^{3/2}-\big(\widehat{\Sigma}_{YY}^{(n)}+\varepsilon_n I\big)^{3/2}\big\}+\big(\widehat{\Sigma}_{YY}^{(n)}-\Sigma_{YY}\big)\Big\} \\ \times\big(\widehat{\Sigma}_{YY}^{(n)}+\varepsilon_n I\big)^{-3/2}\widehat{\Sigma}_{YX}^{(n)}(\widehat{\Sigma}_{XX}^{(n)}+\varepsilon_n I)^{-1/2}.$$

From $\big\|(\widehat{\Sigma}_{YY}^{(n)}+\varepsilon_n I)^{-1/2}\big\|\le\frac{1}{\sqrt{\varepsilon_n}}$, $\big\|\big(\widehat{\Sigma}_{YY}^{(n)}+\varepsilon_n I\big)^{-1/2}\widehat{\Sigma}_{YX}^{(n)}\big(\widehat{\Sigma}_{XX}^{(n)}+\varepsilon_n I\big)^{-1/2}\big\|\le 1$, and Lemma 8 in the Appendix, the norm of the above operator is bounded from above by

$$\frac{1}{\varepsilon_n}\Big\{\frac{3}{\sqrt{\varepsilon_n}}\max\big\{\|\Sigma_{YY}+\varepsilon_n I\|^{3/2},\|\widehat{\Sigma}_{YY}^{(n)}+\varepsilon_n I\|^{3/2}\big\}+1\Big\}\|\widehat{\Sigma}_{YY}^{(n)}-\Sigma_{YY}\|.$$

A similar bound also applies to the third term of Eq. (13). An upper bound on the second term of Eq. (13) is $\frac{1}{\varepsilon_n}\|\Sigma_{YX}-\widehat{\Sigma}_{YX}^{(n)}\|$. Thus, the proof is completed using $\|\widehat{\Sigma}_{XX}^{(n)}\|=\|\Sigma_{XX}\|+o_p(1)$, $\|\widehat{\Sigma}_{YY}^{(n)}\|=\|\Sigma_{YY}\|+o_p(1)$, and Lemma 5. ■

In the next theorem, the compactness assumption on V_{YX} plays an essential role.

Lemma 7 *Assume V_{YX} is compact. Then, for a sequence $\varepsilon_n\to 0$,*

$$\big\|(\Sigma_{YY}+\varepsilon_n I)^{-1/2}\Sigma_{YX}(\Sigma_{XX}+\varepsilon_n I)^{-1/2}-V_{YX}\big\|\;\to\;0\quad(n\to\infty).$$

Proof An upper bound of the left hand side of the assertion is given by

$$\big\|\big\{(\Sigma_{YY}+\varepsilon_n I)^{-1/2}-\Sigma_{YY}^{-1/2}\big\}\Sigma_{YX}(\Sigma_{XX}+\varepsilon_n I)^{-1/2}\big\| \\ +\big\|\Sigma_{YY}^{-1/2}\Sigma_{YX}\big\{(\Sigma_{XX}+\varepsilon_n I)^{-1/2}-\Sigma_{XX}^{-1/2}\big\}\big\|. \quad (14)$$

The first term of Eq. (14) is upper bounded by

$$\big\|\big\{(\Sigma_{YY}+\varepsilon_n I)^{-1/2}\Sigma_{YY}^{1/2}-I\big\}V_{YX}\big\|. \quad (15)$$

Note that the range of V_{YX} is included in $\overline{\mathcal{R}(\Sigma_{YY})}$, as pointed out in Section 2.2. Let v be an arbitrary element in $\mathcal{R}(V_{YX})\cap\mathcal{R}(\Sigma_{YY})$. Then there exists $u\in\mathcal{H}_\mathcal{Y}$ such that $v=\Sigma_{YY}u$. Noting that Σ_{YY} and $(\Sigma_{YY}+\varepsilon_n I)^{1/2}$ are commutative, we have

$$\begin{aligned}&\big\|\big\{(\Sigma_{YY}+\varepsilon_n I)^{-1/2}\Sigma_{YY}^{1/2}-I\big\}v\big\|_{\mathcal{H}_\mathcal{Y}}\\ &=\big\|\big\{(\Sigma_{YY}+\varepsilon_n I)^{-1/2}\Sigma_{YY}^{1/2}-I\big\}\Sigma_{YY}u\big\|_{\mathcal{H}_\mathcal{Y}}\\ &=\big\|(\Sigma_{YY}+\varepsilon_n I)^{-1/2}\Sigma_{YY}^{1/2}\big\{\Sigma_{YY}^{1/2}-(\Sigma_{YY}+\varepsilon_n I)^{1/2}\big\}\Sigma_{YY}^{1/2}u\big\|_{\mathcal{H}_\mathcal{Y}}\\ &\le\big\|\Sigma_{YY}^{1/2}-(\Sigma_{YY}+\varepsilon_n I)^{1/2}\big\|\,\big\|\Sigma_{YY}^{1/2}u\big\|_{\mathcal{H}_\mathcal{Y}}.\end{aligned}$$

Since $\Sigma_{YY}+\varepsilon_n I\to\Sigma_{YY}$ in norm means $(\Sigma_{YY}+\varepsilon_n I)^{1/2}\to\Sigma_{YY}^{1/2}$ in norm, the convergence

$$\big\{(\Sigma_{YY}+\varepsilon_n I)^{-1/2}\Sigma_{YY}^{1/2}-I\big\}v\;\to\;0\qquad(n\to\infty) \quad (16)$$

holds for all $v \in \mathcal{R}(V_{YX}) \cap \mathcal{R}(\Sigma_{YY})$. Because V_{YX} is compact, Lemma 9 in the Appendix shows Eq. (15) converges to zero. The convergence of the second term in Eq. (14) can be proved similarly. ■

Note that the assertion of the above theorem does not necessarily hold without the compactness assumption. In fact, if $Y = X$ and the RKHS is infinite dimensional, $V_{YX} = I$ is not compact, and the norm in the left hand of the assertion is $\|\Sigma_{XX}(\Sigma_{XX} + \varepsilon_n I)^{-1} - I\|$. Since Σ_{XX} has arbitrarily small positive eigenvalues, it is easy to see that this norm is equal to one for all n.

5.3 Proof of the Main Theorems

We are now in a position to prove Theorems 1 and 2.

Proof of Theorem 1 From Lemmas 6 and 7, $\widehat{V}_{YX}^{(n)}$ converges to V_{YX} in norm. Because ϕ and ψ are the eigenfunctions corresponding to the largest eigenvalue of $V_{YX}V_{XY}$ and $V_{XY}V_{YX}$, respectively, and a similar fact holds for $\widehat{\phi}_n$ and $\widehat{\psi}_n$, the assertion is obtained by Lemma 10 in Appendix. ■

Proof of Theorem 2 We show only the convergence of $\widehat{f}_n$. Without loss of generality, we can assume $\widehat{\phi}_n \to \phi$ in $\mathcal{H}_X$. The squared $L_2(P_X)$ distance between $\widehat{f}_n - E_X[\widehat{f}_n(X)]$ and $f - E_X[f(X)]$ is given by

$$\|\Sigma_{XX}^{1/2}(\widehat{f}_n - f)\|_{\mathcal{H}_X}^2 = \|\Sigma_{XX}^{1/2}\widehat{f}_n\|_{\mathcal{H}_X}^2 - 2\langle\phi, \Sigma_{XX}^{1/2}\widehat{f}_n\rangle_{\mathcal{H}_X} + \|\phi\|_{\mathcal{H}_X}^2.$$

Thus, it suffices to show $\Sigma_{XX}^{1/2}\widehat{f}_n$ converges to $\phi \in \mathcal{H}_X$ in probability. We have

$$\begin{aligned}\|\Sigma_{XX}^{1/2}\widehat{f}_n - \phi\|_{\mathcal{H}_X} \le\ & \|\Sigma_{XX}^{1/2}\{(\widehat{\Sigma}_{XX}^{(n)} + \varepsilon_n I)^{-1/2} - (\Sigma_{XX} + \varepsilon_n I)^{-1/2}\}\widehat{\phi}_n\|_{\mathcal{H}_X} \\ & + \|\Sigma_{XX}^{1/2}(\Sigma_{XX} + \varepsilon_n I)^{-1/2}(\widehat{\phi}_n - \phi)\|_{\mathcal{H}_X} \\ & + \|\Sigma_{XX}^{1/2}(\Sigma_{XX} + \varepsilon_n I)^{-1/2}\phi - \phi\|_{\mathcal{H}_X}. \qquad (17)\end{aligned}$$

Using the same argument as in the bound of the first term of Eq. (13), the first term in Eq. (17) is shown to converge to zero. The second term obviously converges to zero. Using the assumption $\phi \in \mathcal{R}(\Sigma_{XX})$, the same argument as in the proof of Eq. (16) in Lemma 7 ensures the convergence of the third term to zero, which completes the proof. ■

With the definition of mean square contingency, Theorem 3 can be proved as follows.

Proof of Theorem 3 Since under the assumptions $E_X[k_{\mathcal{X}}(X,X)] < \infty$ and $E_Y[k_{\mathcal{Y}}(Y,Y)] < \infty$ the operators Σ_{XX} and Σ_{YY} are compact and self-adjoint, there exist complete orthonormal systems $\{\varphi_i\}_{i=1}^{\infty}$ and $\{\psi_i\}_{i=1}^{\infty}$ for $\mathcal{H}_X$ and $\mathcal{H}_{\mathcal{Y}}$, respectively, such that $\langle\varphi_j, \Sigma_{XX}\varphi_i\rangle_{\mathcal{H}_X} = \lambda_i\delta_{ij}$ and $\langle\psi_j, \Sigma_{YY}\psi_i\rangle_{\mathcal{H}_{\mathcal{Y}}} = \nu_i\delta_{ij}$, where λ_i and ν_i are nonnegative eigenvalues and δ_{ij} is Kronecker's delta. Let $\tilde{\phi}_i = (\varphi_i - E_X[\varphi_i(X)])/\sqrt{\lambda_i}$ and $\tilde{\psi}_i = (\psi_i - E_Y[\psi_i(Y)])/\sqrt{\nu_i}$. It follows that $(\tilde{\phi}_i, \tilde{\phi}_j)_{L^2(P_X)} = \delta_{ij}$ and $(\tilde{\psi}_i, \tilde{\psi}_j)_{L^2(P_Y)} = \delta_{ij}$, where $(\cdot,\cdot)_{L^2(P_X)}$ and $(\cdot,\cdot)_{L^2(P_Y)}$ denote the inner product of $L^2(P_X)$ and $L^2(P_Y)$, respectively. We have

$$\begin{aligned}\sum_{i,j=1}^{\infty}\langle\psi_j,\Sigma_{YY}^{-1/2}\Sigma_{YX}\Sigma_{XX}^{-1/2}\varphi_i\rangle_{\mathcal{H}_\mathcal{Y}}^2 &= \sum_{i,j=1}^{\infty}\Big\langle\frac{\psi_i}{\sqrt{\nu_j}},\Sigma_{YX}\frac{\varphi_i}{\sqrt{\lambda_i}}\Big\rangle_{\mathcal{H}_\mathcal{Y}}^2\\ &= \sum_{i,j=1}^{\infty}E_{XY}[\tilde\phi_i(X)\tilde\psi_j(Y)]^2.\end{aligned}$$

Note that we do not need to consider the eigenfunctions with respect to the zero eigenvalue in the sum, because $\overline{\mathcal{R}(V_{YX})}=\overline{\mathcal{R}(\Sigma_{YY})}=\mathcal{N}(\Sigma_{YY})^\perp$ and $\mathcal{N}(V_{YX})=\mathcal{N}(\Sigma_{XX})$.

Since the set $\{\tilde\phi_i\tilde\psi_j\}$ is orthonormal in $L^2(P_X\times P_Y)$, we obtain

$$\begin{aligned}\sum_{i,j=1}^{\infty}E_{XY}[\tilde\phi_i(X)\tilde\psi_j(Y)]^2 &= \sum_{i,j=1}^{\infty}\Big\{\int\int\frac{p_{XY}(x,y)}{p_X(x)p_Y(y)}\tilde\phi_i(x)\tilde\psi_j(y)dP_XdP_Y\Big\}^2\\ &\le \|\zeta+1\|_{L^2(P_X\times P_Y)}^2,\end{aligned}$$

which is finite by assumption. ■

6. Concluding Remarks

We have established the statistical convergence of kernel CCA and NOCCO, showing that the finite sample estimators of the relevant nonlinear mappings converge to the desired population functions. This convergence is proved in the RKHS norm for NOCCO, and in the L_2 norm for kernel CCA. These results give a theoretical justification for using the empirical estimates of NOCCO and kernel CCA in practice.

We have also derived a sufficient condition, $n^{1/3}\varepsilon_n\to\infty$, for the decay of the regularization coefficient ε_n, which ensures the convergence described above. As Leurgans et al. (1993) suggest, the order of the sufficient condition seems to depend on the functional norm used to determine convergence. An interesting question is whether the theoretical order $n^{1/3}\varepsilon_n\to\infty$ can be improved for convergence in the L_2 or RKHS norm.

A result relevant to the convergence of kernel principal component analysis (KPCA) has recently been obtained by Zwald and Blanchard (2006). They show a probabilistic upper bound on the difference between the projectors onto the D-dimensional population eigenspaces and the empirical eigenspaces. Since KPCA needs no inversion operation, the theoretical analysis is easier than for kernel CCA. That said, it would be very interesting to consider the applicability of the methods developed by Zwald and Blanchard (2006) to kernel CCA.

There are some practical problems that remain to be addressed when applying kernel CCA and related methods. One of the problems is how to choose the regularization coefficient ε_n in practice. As the numerical simulations in Section 4 show, the order $n^{-1/3}$ is only a sufficient condition for convergence in general cases, and the optimal ε_n to estimate the true functions may depend on statistical properties of the given data, such as spectral distribution of Gram matrices. This problem should be studied more in future to make the methods more applicable.

The choice of kernel is another important unsolved problem. The kernel defines the meaning of ”nonlinear correlation” through an assumed class of functions, and thus determines how to measure the dependence structure of the data. If a parameterized family of kernels such as the Gaussian RBF kernel is provided, then cross-validation might be a reasonable method to select the best kernel (see Leurgans et al., 1993), however this remains to be established.

One of the methods related to kernel CCA is independent component analysis (ICA), since Bach and Jordan (2002) use kernel CCA in their kernel ICA algorithm. The theoretical results developed in this paper will work as a basis for analyzing the properties of the kernel ICA algorithm; in particular, for demonstrating statistical consistency of the estimator. Since ICA estimates the demixing matrix as a parameter, however, we need to consider covariance operators parameterized by this matrix, and must discuss how convergence of the objective function depends on the parameter. It is not a straightforward task to obtain consistency of kernel ICA from the results of this paper. Extending our results to the parametric case is an interesting topic for future work.

Acknowledgments

The first author was supported by JSPS KAKENHI 15700241, Research Grant of Inamori Foundation, Scientific Grant of Mitsubishi Foundation, and Transdisciplinary Research Integration Projects of ROIS. This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. The authors thank Dr. Shotaro Akaho, Yukito Iba, Hiromichi Suetani, and Matthias Hein for their helpful comments, and thank editor and anonymous reviewers for their comments to improve the manuscript. Part of this work was done while the first author was visiting Max Planck Institute for Biological Cybernetics.

Appendix A. Basics from Functional Analysis

We briefly give definitions and basic properties of compact and Hilbert-Schmidt operators. For complete references, see, for example, Reed and Simon (1980), Dunford and Schwartz (1963), and Lax (2002), among others.

Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be Hilbert spaces. A bounded operator $T : \mathcal{H}_1 \to \mathcal{H}_2$ is called *compact* if for every bounded sequence $\{f_n\} \subset \mathcal{H}_1$ the image $\{Tf_n\}$ has a subsequence which converges in $\mathcal{H}_2$. By the Heine-Borel theorem, finite rank operators are necessarily compact. Among many useful properties of compact operators, singular value decomposition is available. For a compact operator $T : \mathcal{H}_1 \to \mathcal{H}_2$, there exist $N \in \mathbb{N} \cup \{\infty\}$, a non-increasing sequence of positive numbers $\{\lambda_i\}_{i=1}^N$, and (not necessarily complete) orthonormal systems $\{\phi_i\}_{i=1}^N \subset \mathcal{H}_1$ and $\{\psi_i\}_{i=1}^N \subset \mathcal{H}_2$ such that

$$T = \sum_{i=1}^{N} \lambda_i \langle \phi_i, \cdot \rangle_{\mathcal{H}_1} \psi_i.$$

If $N = \infty$, then $\lambda_i \to 0$ $(i \to \infty)$ and the infinite series in the above equation converges in norm.

Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be Hilbert spaces. A bounded operator $T : \mathcal{H}_1 \to \mathcal{H}_2$ is called *Hilbert-Schmidt* if $\sum_{i=1}^\infty \|T\varphi_i\|_{\mathcal{H}_2}^2 < \infty$ for a CONS $\{\varphi_i\}_{i=1}^\infty$ of $\mathcal{H}_1$. It is known that this sum is independent of the choice of a CONS. For two Hilbert-Schmidt operators T_1 and T_2, the Hilbert-Schmidt inner product is defined by

$$\langle T_1, T_2 \rangle_{HS} = \sum_{i=1}^{\infty} \langle T_1\varphi_i, T_2\varphi_i \rangle_{\mathcal{H}_2},$$

with which the set of all Hilbert-Schmidt operators from $\mathcal{H}_1$ to $\mathcal{H}_2$ is a Hilbert space. The Hilbert-Schmidt norm $\|T\|_{HS}$ is defined by $\|T\|_{HS}^2 = \langle T, T \rangle_{HS} = \sum_{i=1}^\infty \|T\varphi_i\|_{\mathcal{H}_2}^2$ as usual. Obviously, for a

Hilbert-Schmidt operator T, we have

$$\|T\| \le \|T\|_{HS}.$$

Appendix B. Lemmas Used in the Proofs

We show three lemmas used in the proofs in Section 5. Although they may be basic facts, we show the complete proofs for convenience.

Lemma 8 *Suppose A and B are positive self-adjoint operators on a Hilbert space such that $0 \le A \le \lambda I$ and $0 \le B \le \lambda I$ hold for a positive constant λ. Then,*

$$\|A^{3/2} - B^{3/2}\| \le 3\lambda^{1/2}\|A - B\|.$$

Proof Without loss of generality we can assume $\lambda = 1$. Define functions f and g on $\{z \mid |z| \le 1\}$ by $f(z) = (1-z)^{3/2}$ and $g(z) = (1-z)^{1/2}$. Let

$$f(z) = \sum_{n=1}^{\infty} b_n z^n \qquad \text{and} \qquad g(z) = \sum_{n=0}^{\infty} c_n z^n$$

be the power series expansions. They converge absolutely for $|z| \le 1$. In fact, because direct differentiation yields $b_0 = 1, b_1 = -\frac{3}{2}$, and $b_n > 0$ for $n \ge 2$, the inequality

$$\begin{aligned}\sum_{n=0}^{N} |b_n| &= 1 + \frac{3}{2} + \sum_{n=2}^{N} b_n = 1 + \frac{3}{2} + \lim_{x \uparrow 1} \sum_{n=2}^{N} b_n x^n \\ &\le 1 + \frac{3}{2} + \lim_{x\uparrow 1}\left\{ f(x) - 1 + \frac{3}{2}\right\} = 3\end{aligned}$$

shows the convergence of $\sum_{n=0}^{\infty} b_n z^n$ for $|z| = 1$. The bound $\sum_{n=0}^{\infty} |c_n| \le 2$ can be proved similarly.

From $0 \le I - A, I - B \le I$, we have $f(I-A) = A^{3/2}$, $f(I-B) = B^{3/2}$, and thus,

$$\|A^{3/2} - B^{3/2}\| = \left\| \sum_{n=0}^{\infty} b_n (I-A)^n - \sum_{n=0}^{\infty} b_n (I-B)^n \right\| \le \sum_{n=0}^{\infty} |b_n| \|(I-A)^n - (I-B)^n\|.$$

It is easy to see $\|T^n - S^n\| \le n\|T - S\|$ by induction for operators T and S with $\|T\| \le 1$ and $\|S\| \le 1$. From $f'(z) = -\frac{3}{2} g(z)$, the relation $nb_n = -\frac{3}{2} c_n$ holds for all n. Therefore, we obtain

$$\|A^{3/2} - B^{3/2}\| \le \sum_{n=0}^{\infty} n|b_n| \|A - B\| = \frac{3}{2} \sum_{n=0}^{\infty} |c_n| \|A - B\| \le 3\|A - B\|.$$

■

The following lemma is a slight extension of Exercise 9, Section 21.2 in Lax (2002).

Lemma 9 *Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be Hilbert spaces, and $\mathcal{H}_0$ be a dense linear subspace of $\mathcal{H}_2$. Suppose A_n and A are bounded operators on $\mathcal{H}_2$, and B is a compact operator from $\mathcal{H}_1$ to $\mathcal{H}_2$ such that*

$$A_n u \to Au$$

for all $u \in \mathcal{H}_0$, and

$$\sup_n \|A_n\| \le M$$

for some $M > 0$. Then $A_n B$ converges to AB in norm.

Proof First, we prove that $A_n u \to Au$ holds for an arbitrary $u \in \mathcal{H}_2$. For any $\varepsilon > 0$, there is $u_0 \in \mathcal{H}_0$ so that $\|u - u_0\|_{\mathcal{H}_2} \le \varepsilon/(2(M + \|A\|))$. For $u_0 \in \mathcal{H}_0$, there is $N \in \mathbb{N}$ such that $\|A_n u_0 - Au_0\|_{\mathcal{H}_2} \le \varepsilon/2$ for all $n \ge N$. Then for all $n \ge N$ we have

$$\|A_n u - Au\|_{\mathcal{H}_2} \le \|A_n\| \|u - u_0\|_{\mathcal{H}_2} + \|A_n u_0 - Au_0\|_{\mathcal{H}_2} + \|A\| \|u - u_0\|_{\mathcal{H}_2} \le \varepsilon.$$

Next, assume that the operator norm $\|A_n B - AB\|$ does not converge to zero. Then there exist $\delta > 0$ and a subsequence (n') such that $\|A_{n'} B - AB\| \ge 2\delta$. For each n' there exists $v_{n'} \in \mathcal{H}_1$ such that $\|v_{n'}\|_{\mathcal{H}_1} = 1$ and $\|A_{n'} B v_{n'} - ABv_{n'}\|_{\mathcal{H}_2} \ge \delta$. Let $u_{n'} = Bv_{n'}$. Because B is compact and $\|v_{n'}\|_{\mathcal{H}_1} = 1$, there is a subsequence $u_{n''}$ and u_* in $\mathcal{H}_2$ such that $u_{n''} \to u_*$. We have

$$\begin{aligned}
&\|A_{n''} u_{n''} - Au_{n''}\|_{\mathcal{H}_2} \\
&\le \|A_{n''}(u_{n''} - u_*)\|_{\mathcal{H}_2} + \|(A_{n''} - A)u_*\|_{\mathcal{H}_2} + \|A(u_{n''} - u_*)\|_{\mathcal{H}_2} \\
&\le (M + \|A\|)\|u_{n''} - u_*\|_{\mathcal{H}_2} + \|(A_{n''} - A)u_*\|_{\mathcal{H}_2},
\end{aligned}$$

which converges to zero as $n'' \to \infty$. This contradicts the choice of $v_{n'}$. ■

Lemma 10 *Let A be a compact positive operator on a Hilbert space $\mathcal{H}$, and A_n ($n \in \mathbb{N}$) be bounded positive operators on $\mathcal{H}$ such that A_n converges to A in norm. Assume that the eigenspace of A corresponding to the largest eigenvalue is one-dimensional spanned by a unit eigenvector ϕ, and the maximum of the spectrum of A_n is attained by a unit eigenvector f_n. Then*

$$|\langle f_n, \phi \rangle_{\mathcal{H}}| \to 1 \quad (n \to \infty).$$

Proof Because A is compact and positive, the eigendecomposition

$$A = \sum_{i=1}^{\infty} \rho_i \phi_i \langle \phi_i, \cdot \rangle_{\mathcal{H}}$$

holds, where $\rho_1 > \rho_2 \ge \rho_3 \ge \cdots \ge 0$ are eigenvalues and $\{\phi_i\}$ is the corresponding eigenvectors so that $\{\phi_i\}$ is the CONS of $\mathcal{H}$.

Let $\delta_n = |\langle f_n, \phi_1 \rangle|$. We have

$$\begin{aligned}
\langle f_n, Af_n \rangle &= \rho_1 \langle f_n, \phi_1 \rangle^2 + \sum_{i=2}^{\infty} \rho_i \langle \phi_i, f_n \rangle^2 \\
&\le \rho_1 \langle f_n, \phi_1 \rangle^2 + \rho_2 \big(1 - \langle f_n, \phi_1 \rangle^2\big) = \rho_1 \delta_n^2 + \rho_2 (1 - \delta_n^2).
\end{aligned}$$

On the other hand, the convergence

$$\begin{aligned}
|\langle f_n, Af_n \rangle - \langle \phi_1, A\phi_1 \rangle| &\le |\langle f_n, Af_n \rangle - \langle f_n, A_n f_n \rangle| + |\langle f_n, A_n f_n \rangle - \langle \phi_1, A\phi_1 \rangle| \\
&\le \|A - A_n\| + \big|\|A_n\| - \|A\|\big| \quad \to \quad 0
\end{aligned}$$

implies that $\langle f_n, Af_n \rangle$ must converges to ρ_1. These two facts, together with $\rho_1 > \rho_2$, result in $\delta_n \to 1$. ■

Note that from the norm convergence $Q_n A_n Q_n \to QAQ$, where Q_n and Q are the orthogonal projections onto the orthogonal complements of ϕ_n and ϕ, respectively, we have convergence of the eigenvector corresponding to the second eigenvalue. It is not difficult to obtain convergence of the eigenspaces corresponding to the m-th eigenvalue in a similar way.

References

Shotaro Akaho. A kernel method for canonical correlation analysis. In *Proceedings of International Meeting on Psychometric Society (IMPS2001)*, 2001.

Theodore W. Anderson. *An Introduction to Multivariate Statistical Analysis*. John Wiley & Sons, third edition, 2003.

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 69(3):337–404, 1950.

Francis R. Bach and Michael I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.

Charles R. Baker. Joint measures and cross-covariance operators. *Transactions of the American Mathematical Society*, 186:273–289, 1973.

Leo Breiman and Jerome H. Friedman. Estimating optimal transformations for multiple regression and correlation. *Journal of the American Statistical Association*, 80:580–598, 1985.

Andreas Buja. Remarks on functional canonical variates, alternating least squares methods and ACE. *The Annals of Statistics*, 18(3):1032–1069, 1990.

Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.

Nelson Dunford and Jacob T. Schwartz. *Linear Operators, Part II*. Interscience, 1963.

Kenji Fukumizu, Francis R. Bach, and Michael I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

Micheal J. Greenacre. *Theory and Applications of Correspondence Analysis*. Academic Press, London, 1984.

Arthur Gretton, Olivier Bousquet, Alexander J. Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In S. Jain, H. U. Simon, and E. Tomita, editors, *16th International Conference on Algorithmic Learning Theory*, pages 63–77. Springer, 2005a.

Arthur Gretton, Alexander J. Smola, Olivier Bousquet, R. Herbrich, A. Belitski, M. Augath, Y. Murayama, J. Pauls, B. Schölkopf, and N. Logothetis. Kernel constrained covariance for dependence measurement. In *AISTATS*, volume 10, 2005b.

Charles W. Groetsch. *The Theory of Tikhonov Regularization for Fredholm Equations of the First Kind*. Pitman, 1984.

David R. Hardoon, John Shawe-Taylor, and Ola Friman. KCCA for fMRI analysis. In *Proceedings of Medical Image Understanding and Analysis (London)*, 2004.

Guozhong He, Hans-Georg Müller, and Jane-Ling Wang. Functional canonical analysis for square integrable stochastic procersses. *Journal of Multivariate Analysis*, 85:54–77, 2003.

Peter D. Lax. *Functional Analysis*. Wiley, 2002.

Sue E. Leurgans, Rana A. Moyeed, and Bernard W. Silverman. Canonical correlation analysis when the data are curves. *Journal of the Royal Statistical Society, Series B*, 55(3):725–740, 1993.

Thomas Melzer, Michael Reiter, and Horst Bischof. Nonlinear feature extraction using generalized canonical correlation analysis. In *Proceedings of International Conference on Artificial Neural Networks (ICANN)*, pages 353–360, 2001.

Sebastian Mika, Gunnar Rätsch, Jason Weston, Bernhard Schölkopf, and Klaus-Robert Müller. Fisher discriminant analysis with kernels. In Y.-H. Hu, J. Larsen, E. Wilson, and S. Douglas, editors, *Neural Networks for Signal Processing*, volume IX, pages 41–48. IEEE, 1999.

Michael Reed and Barry Simon. *Functional Analysis*. Academic Press, 1980.

Alfréd Rényi. *Probability Theory*. Horth-Holland, 1970.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. MIT Press, 2002.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

Ingo Steinwart, Don Hush, and Clint Scovel. An explicit description of the reproducing kernel Hilbert spaces of Gaussian RBF kernels. Technical Report LA-UR-04-8274, Los Alamos National Laboratory, 2004.

Hiromichi Suetani, Yukito Iba, and Kazuyuki Aihara. Detecting hidden synchronization of chaotic dynamical systems: A kernel-based approach. *Journal of Physics A: Mathematical and General*, 39:10723–10742, 2006.

Nikolai N. Vakhania, Vazha I. Tarieladze, and Sergei A. Chobanyan. *Probability Distributions on Banach Spaces*. D. Reidel Publishing Company, 1987.

Ard W. van der Vaart. *Asymptotic Statistics*. Cambridge University Press, 1998.

Yoshihiro Yamanishi, Jean-Philippe Vert, Akihiro Nakaya, and Minoru Kanehisa. Extraction of correlated gene clusters from multiple genomic data by generalized kernel canonical correlation analysis. *Bioinformatics*, 19:323i–330i, 2003.

Laurent Zwald and Gilles Blanchard. On the convergence of eigenspaces in kernel principal component analysis. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*. MIT Press, Cambridge, MA, 2006.

Journal of Machine Learning Research 8 (2007) 385-408 Submitted 6/06; Revised 12/06; Published 3/07

Learning Equivariant Functions with Matrix Valued Kernels

Marco Reisert REISERT@INFORMATIK.UNI-FREIBURG.DE
Hans Burkhardt BURKHARDT@INFORMATIK.UNI-FREIBURG.DE
LMB, Georges-Koehler-Allee 52
Albert-Ludwig University
79110 Freiburg, Germany

Editor: Leslie Pack Kaelbing

Abstract

This paper presents a new class of matrix valued kernels that are ideally suited to learn vector valued equivariant functions. Matrix valued kernels are a natural generalization of the common notion of a kernel. We set the theoretical foundations of so called equivariant matrix valued kernels. We work out several properties of equivariant kernels, we give an interpretation of their behavior and show relations to scalar kernels. The notion of (ir)reducibility of group representations is transferred into the framework of matrix valued kernels. At the end to two exemplary applications are demonstrated. We design a non-linear rotation and translation equivariant filter for 2D-images and propose an invariant object detector based on the generalized Hough transform.

Keywords: kernel methods, matrix kernels, equivariance, group integration, representation theory, Hough transform, signal processing, Volterra series

1. Introduction

In the last decade kernel techniques have gained much attention in machine learning theory. There is a large variety of problems which can be naturally formulated in a kernelized manner, ranging from regression and classification problems, over feature transformation algorithms to coding schemes. There is a large literature on this subject. We recommend Schoelkopf and Smola (2002) and Shawe-Taylor and Cristianini (2004) and references therein.

The notion of a kernel as a kind of similarity between two given patterns can be naturally generalized to matrix valued kernels. A matrix valued kernel can carry more information than only similarity, like information about the relative pose or configuration of the two patterns. It is one way to overcome the 'information bottleneck' of the kernel matrix. First Burbea and Masani (1984) studied the theory of vector-valued reproducing kernel Hilbert spaces leading to the notion of a matrix (or operator) valued kernel. Amodei (1996) applied it for the solution of Partial Differential Equations and more recently Micchelli and Pontil (2005) used it in the context of machine learning.

In this paper we introduce a new class of matrix valued kernels with a specific transformation behavior. The new type of kernel is motivated by a vector valued regression problem. The function to be learned is desired to be equivariant, meaning that group actions on the input space of the function translate to corresponding group actions on the output space. To get an intuition: from signal processing theory one is familiar with the notion of a 'time-invariant linear filter'. In our words the term 'time-invariant' means that the filter is equivariant to the group of time-shifts. More exactly, the time-shift equivariance is expressed by the fact that the group-representations of time-shifts and the action of the filter (in this case a linear convolution) commute. Equivariance is of high interest in signal-/image processing problems and geometrically related learning problems.

We give a constructive way to obtain kernels, whose linear combinations lead to equivariant functions. These kernels are based on the concept of group integration. Group integration is widely used for invariant feature extraction. For a introduction see Burkhardt and Siggelkow (2001). Recently Haasdonk et al. (2005) used group integration to get invariance in kernel methods.

The paper is organized as follows: In Section 2 we give a first motivation for our kernels by solving an equivariant regression problem. We give an illustrative interpretation of the kernels and present an equivariant Representer Theorem, which justifies our lax motivation from the beginning. Further we uncover a relation of our framework to Volterra theory. In Section 3 we give rules how to construct matrix kernels out of matrix kernels and give constraints how to preserve equivariance. Section 4 introduces the notion of irreducibility for kernels and show how the traces of matrix valued kernels are related to scalar valued kernels. Two possible application of matrix valued kernels are given in Section 5: a rotation equivariant non-linear filter and a rotation invariant object detector. Finally, Section 6 gives a conclusion and an outlook for future direction of research and applications.

2. Group Integration Matrix Kernels

In this section we propose the basic approach. After some preliminaries we give a first motivation for our idea. Then we give an interpretation of the proposed kernels and present an equivariant Representer Theorem.

2.1 Preliminaries and Notation

We consider compact (including finite), linear, unimodular groups $\mathcal{G}$. A group representation ρ_g is a group homomorphism $\mathcal{G} \mapsto L(\mathcal{V})$, where $\mathcal{V}$ is a finite dimensional Hilbert space usually called the representation space. Sometimes we write $g\mathbf{x}$ meaning $\rho_g \mathbf{x}$, where the patterns $\mathbf{x} \in \mathcal{V}$ are always in bold face. In the continuous case ($\mathcal{G}$ is a Lie group), the representation should also be continuous. In this case one can show that any representation of the group is equivalent to an unitary representation. For finite groups this statement also holds (for a proof see, for example, Lenz, 1990). So we can restrict ourselves with no loss of generality to unitary group representation, that is, always $\rho_{g^{-1}} = \rho_g^\dagger$ holds. We denote the Haar Integral (or Group Integral) by $\int_{\mathcal{G}} f(g)\, dg$ regardless whether we deal with finite groups or continuous groups. Due to the unimodularity all reparametrizations are possible. For further reading on these topics we recommend Lenz (1990), Gaal (1973), Nachbin (1965), and Miller (1991); Schur (1968).

We want to learn functions $\mathbf{f} : \mathcal{X} \mapsto \mathcal{Y}$, where $\mathcal{X}, \mathcal{Y}$ are finite dimensional Hilbert spaces. Tensor- or Kronecker products are denoted by $\otimes$. An equivariant function is a function fulfilling $\mathbf{f}(g\mathbf{x}) = g\mathbf{f}(\mathbf{x})$. The group actions on $\mathcal{X}$ and $\mathcal{Y}$ can be chosen such that they apply to the current application or problem. Inner products are denoted by $\langle \cdot | \cdot \rangle_{\mathcal{X}}$, where the subscript indicates in which space the arguments are living. Due to the unitary group representation the inner product is invariant in the sense $\langle g\mathbf{x}_1 | g\mathbf{x}_2 \rangle_{\mathcal{X}} = \langle \mathbf{x}_1 | \mathbf{x}_2 \rangle_{\mathcal{X}}$. A scalar valued kernel is a symmetric, positive definite function $k : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ fulfilling the Mercer property. More precisely, we use the term positive definite in the sense of semi-positive definiteness. If strict positive definiteness is forced, we make this explicit. We also assume invariance to group actions of $\mathcal{G}$, that is, $k(g\mathbf{x}_1, g\mathbf{x}_2) = k(\mathbf{x}_1, \mathbf{x}_2)$ for all $g \in \mathcal{G}$. The function space spanned by all vector-valued linear combination of kernel evaluations of k is called

$\mathcal{Z}_k$ (isomorphic to the vector-valued RKHS of k), where the vector-valued coefficients live in $\mathcal{Y}$, that is, $\mathcal{Z}_k = \{\mathbf{f}(\mathbf{x}) = \sum_i k(\mathbf{x},\mathbf{x}_i)\mathbf{a}_i \mid \mathbf{x}_i \in \mathcal{X},\ \mathbf{a}_i \in \mathcal{Y}\}$.

We assume the reader is familiar with basics in kernel methods, tensor algebra and representation theory.

2.2 Motivation

Our goal is to learn functions $\mathbf{f}: \mathcal{X} \mapsto \mathcal{Y}$ from learning samples $\{(\mathbf{x}_i,\mathbf{y}_i)|i = 1..n\}$ in an equivariant manner, that is, the function has to satisfy $\mathbf{f}(g\mathbf{x}) = g\mathbf{f}(\mathbf{x})$ for all $g \in \mathcal{G}$, while $\mathbf{f}(\mathbf{x}_i) = \mathbf{y}_i$ should be fulfilled as accurate as possible.

Due to the Representer Theorem by Kimeldorf and Wahba (1971) minimizer in $\mathcal{Z}_k$ of arbitrary risk functionals are of the form

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{n} k(\mathbf{x},\mathbf{x}_i)\mathbf{a}_i, \tag{1}$$

where the $\mathbf{a}_i \in \mathcal{Y}$ are some vector valued coefficients, which have to be learned. To obtain an equivariant behavior we pretend to present the training samples in all possible poses $\{(g\mathbf{x}_i, g\mathbf{y}_i)|i = 1..n,\ g \in \mathcal{G}\}$. Since no pose g should be favored, the coefficients $\mathbf{a}_i$ are not allowed to depend on g explicitly, but they have to turn its pose according to the actions in the input space. And hence the solution has to look like

$$\mathbf{f}(\mathbf{x}) = \int_{\mathcal{G}} \sum_{i=1}^{n} k(\mathbf{x}, g\mathbf{x}_i)\ g\mathbf{a}_i\ dg, \tag{2}$$

where the integral ranges over the whole group $\mathcal{G}$. As desired the following lemma holds

Lemma 2.1 *Every function of form (2) is an equivariant function with respect to $\mathcal{G}$.*

Proof Due to the invariance of the kernel we have

$$\mathbf{f}(g\mathbf{x}) = \int_{\mathcal{G}} \sum_{i} k(\mathbf{x}, g^{-1}g'\mathbf{x}_i)\ g'\mathbf{a}_i\ dg',$$

and using the unimodularity of $\mathcal{G}$ we reparametrize the integral by $h = g^{-1}g'$ and obtain the desired result

$$\mathbf{f}(g\mathbf{x}) = \int_{\mathcal{G}} \sum_{i} k(\mathbf{x}, h\mathbf{x}_i)\ gh\mathbf{a}_i\ dh = g\mathbf{f}(\mathbf{x}),$$

using the linearity of the group representation. ■

So we have a constructive way to obtain equivariant functions. Since we can exchange summation with integration and the group action is linear we can rewrite (2) by

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{n} \left(\int_{\mathcal{G}} k(\mathbf{x}, g\mathbf{x}_i)\ \rho_g\ dg \right) \mathbf{a}_i,$$

where we can interpret the expression inside the brackets as a matrix valued kernel. First we want to characterize its basic properties in the following

Proposition 2.2 *Let $K : \mathcal{X} \times \mathcal{X} \mapsto L(\mathcal{Y})$ for every $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$ be defined by*

$$K(\mathbf{x}_1, \mathbf{x}_2) = \int_{\mathcal{G}} k(\mathbf{x}_1, g\mathbf{x}_2)\ \rho_g\ dg,$$

where $\rho_g \in L(\mathcal{Y})$ is a group representation and k is any symmetric, $\mathcal{G}$-invariant function. The following properties hold for a function above,

a) For every $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$ and $g, h \in \mathcal{G}$, we have that

$$K(g\mathbf{x}_1, h\mathbf{x}_2) = \rho_g\ K(\mathbf{x}_1, \mathbf{x}_2)\ \rho_h^\dagger,$$

that is, K is equivariant in the first argument and anti-equivariant in the second, we say K is equivariant.

b) It holds $K(\mathbf{x}_1, \mathbf{x}_2) = K(\mathbf{x}_2, \mathbf{x}_1)^\dagger$.

Proof To prove a) we just have to follow the reasoning in the proof of Lemma 2.1. For b) we use the invariance and symmetry of k and the unimodularity of $\mathcal{G}$ and get

$$K(\mathbf{x}_1, \mathbf{x}_2) \quad = \quad \int_{\mathcal{G}} k(g^{-1}\mathbf{x}_1, \mathbf{x}_2)\ \rho_g\ dg = \int_{\mathcal{G}} k(\mathbf{x}_2, g\mathbf{x}_1)\ \rho_{g^{-1}}\ dg = K(\mathbf{x}_2, \mathbf{x}_1)^\dagger,$$

as asserted. ■

There are basically two ways to introduce matrix valued kernels in almost the same manner as for scalar-valued kernels. Following Aronszajn (1950) one can introduce a vector valued reproducing kernel Hilbert space (RKHS) and then implicitly assume that there exists a kind of evaluation functional $K_\mathbf{x}$ in a certain Hilbert space of vector valued functions (for a more recent introduction in the context of matrix valued kernels see Pontil and Micchelli (2004)). Basically the approach is as follows: by the Riesz Lemma there is a $K_\mathbf{x}$ such that $\langle \mathbf{y} | \mathbf{f}(\mathbf{x}) \rangle_{\mathcal{Y}} = \langle K_\mathbf{x} \mathbf{y} | \mathbf{f} \rangle$, where the inner product on the right hand side works in the Hilbert space of vector valued functions. As the upper expression is linear in $\mathbf{y}$ one can define a linear operator in $\mathcal{Y}$, the so called matrix valued kernel $K(\mathbf{x}_1, \mathbf{x}_2) \in L(\mathcal{Y})$, by the following $K(\mathbf{x}_1, \mathbf{x}_2)\mathbf{y} := (K_{\mathbf{x}_2}\mathbf{y})(\mathbf{x}_1)$ equation.

A second possibility to introduce the notion of a matrix valued kernel is to demand the existence of a feature mapping into a certain feature space. For example for scalar valued kernels this is done by Shawe-Taylor and Cristianini (2004). For matrix valued kernels the feature space can be identified as the tensor product space of linear mappings on $\mathcal{Y}$ times an arbitrary high dimensional feature space.

Both, the RKHS approach and the feature space approach are equivalent due to a generalized Mercer Theorem. We decided for the latter alternative. It is more appropriate for our purposes, because we can explicitly access the structure of the induced feature space.

Definition 2.3 (Matrix Valued Kernel) *A function $K : \mathcal{X} \times \mathcal{X} \mapsto L(\mathcal{Y})$ is called a matrix valued kernel if there is a sesqui-linear form such that for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$*

$$K(\mathbf{x}_1, \mathbf{x}_2) = \langle \Psi(\mathbf{x}_1) | \Psi(\mathbf{x}_2) \rangle_{\mathcal{H}}$$

holds, where Ψ is a mapping from $\mathcal{X}$ into a feature-space $\mathcal{H} = \mathcal{F} \otimes L(\mathcal{Y})$ with the property that for all $0 \neq \Psi \in \mathcal{H}$ the matrix $\langle \Psi | \Psi \rangle_{\mathcal{H}} \in L(\mathcal{Y})$ is positive definite.

The elements of the feature-space $\mathcal{H}$ can be imagined as $L(\mathcal{Y})$-valued or matrix-valued vectors, just elements of $\mathcal{F} \otimes L(\mathcal{Y})$. Actually it carries the structure of a $L(\mathcal{Y})$-bimodule. A module is a generalization of a vector space where the field elements are replaced by noncommutative ring elements. For a bimodule right and left multiplication are defined (see, for example, R. G. Douglas, 1989). Additionally we have to define the adjoint for the ring-elements $L(\mathcal{Y})$, which simply turns out to be the adjoint of the linear mapping. The connection of the feature space representation to the RKHS of Aronszjan is simple. Similarly to the case of the scalar kernels, the evaluation functionals $K_\mathbf{x}$ have to be associated with the elements $\Psi(\mathbf{x})$ living in the feature space $\mathcal{F} \otimes L(\mathcal{Y})$. In the RKHS the functions are represented by linear combinations $\sum_i K_{\mathbf{x}_i} \mathbf{a}_i$; in the feature space the linear combination takes the form $\sum_i \Psi(\mathbf{x}_i)\mathbf{a}_i$ which is an element of $\mathcal{F} \otimes \mathcal{Y}$. The space $\mathcal{F} \otimes \mathcal{Y}$ carries the structure of a $L(\mathcal{Y})$-left module, because multiplication from the left by elements in $L(\mathcal{Y})$ is allowed. Note that this a fundamental difference to scalar kernels. In the case of scalar kernels the representation of functions and the evaluation functionals live in the same space. For matrix kernels it is different. The evaluation functional are elements of a $L(\mathcal{Y})$-bimodule or explicitly $\mathcal{F} \otimes L(\mathcal{Y})$; the functions are elements of the $L(\mathcal{Y})$-left module $\mathcal{F} \otimes \mathcal{Y}$.

Remark 2.4 *In Definition 2.3 the matrix-valued inner product in $\mathcal{H}$ also induces a scalar inner product by the trace $tr(\langle \Psi | \Psi' \rangle_\mathcal{H})$ and it indeed holds that $tr(\langle \Psi | \Psi \rangle_\mathcal{H}) \geq 0$ for all $\Psi \neq 0$. Hence the space $\mathcal{H}$ is naturally attached with a (semi-)norm $||\Psi|| = \sqrt{tr(\langle \Psi | \Psi \rangle_\mathcal{H})}$.*

Theorem 2.5 (GIM-kernels) *If the function k is a scalar kernel, then the function K given in Proposition 2.2 is a matrix valued kernel. We call this kernel a Group Integration Matrix kernel or short GIM-kernel.*

Proof To show this, we give the feature mapping Ψ by the use of the feature mapping Φ corresponding to the scalar kernel k and show that the GIM-kernel can be written as a positive matrix valued inner product in the new feature space $\mathcal{H} = \mathcal{F} \otimes L(\mathcal{Y})$. Let Ψ be given by

$$\Psi(\mathbf{x}) = \frac{1}{\sqrt{\mu(\mathcal{G})}} \int_\mathcal{G} \Phi(g\mathbf{x}) \otimes \rho_g \, dg,$$

where $\mu(\mathcal{G}) = \int_\mathcal{G} dg$ is the volume of the group. The matrix valued inner product in $\mathcal{H}$ is given by the rule

$$\langle \Phi_1 \otimes \rho_1 | \Phi_2 \otimes \rho_2 \rangle_\mathcal{H} := \rho_1^\dagger \rho_2 \langle \Phi_1 | \Phi_2 \rangle_\mathcal{F}$$

and its linear extension, that is, it is a sesqui-linear mapping of type $\mathcal{H} \times \mathcal{H} \mapsto L(\mathcal{Y})$. And the so defined product is indeed positive. Any $\Psi \in \mathcal{H}$ can be written as a sum (integral) of Kronecker products $\Psi = \sum_i \Phi_i \otimes \rho_i$. So for any $\Psi \in \mathcal{H}$ and any $\mathbf{y} \in \mathcal{Y}$ we have

$$\langle \mathbf{y} | \langle \Psi | \Psi \rangle_\mathcal{H} \mathbf{y} \rangle_\mathcal{Y} = \sum_{i,j} \langle \Phi_i | \Phi_j \rangle_\mathcal{F} \langle \rho_i \mathbf{y} | \rho_j \mathbf{y} \rangle_\mathcal{Y} = \sum_{i,j} k_{ij} \langle \rho_i \mathbf{y} | \rho_j \mathbf{y} \rangle_\mathcal{Y} = \sum_{i,j} k_{ij} \langle \mathbf{y}_i | \mathbf{y}_j \rangle_\mathcal{Y} \geq 0$$

is positive, because the matrix k_{ij} is positive definite by the Mercer property of k.

Using the above rule for the inner product we can compute

$$\begin{aligned} \langle \Psi(\mathbf{x}_1) | \Psi(\mathbf{x}_2) \rangle_\mathcal{H} &= \frac{1}{\mu(\mathcal{G})} \int_{\mathcal{G}^2} \langle \Phi(g\mathbf{x}_1) \otimes \rho_g | \Phi(h\mathbf{x}_2) \otimes \rho_h \rangle_\mathcal{H} \, dg \, dh \\ &= \frac{1}{\mu(\mathcal{G})} \int_{\mathcal{G}^2} \langle \Phi(g\mathbf{x}_1) | \Phi(h\mathbf{x}_2) \rangle_\mathcal{F} \, \rho_{g^{-1}h} \, dg \, dh. \end{aligned}$$

Inserting the scalar kernel k and reparametrizing by $g' = g^{-1}h$ gives

$$\langle \Psi(\mathbf{x}_1) | \Psi(\mathbf{x}_2) \rangle_{\mathcal{H}} = \frac{1}{\mu(\mathcal{G})} \int_{\mathcal{G}^2} k(\mathbf{x}_1, g'\mathbf{x}_2) \rho_{g'} \, dg' \, dh = \int_{\mathcal{G}} k(\mathbf{x}_1, g\mathbf{x}_2) \rho_g \, dg = K(\mathbf{x}_1, \mathbf{x}_2)$$

which is the desired result. ■

2.3 Examples and Interpretation of Equivariant Kernels

To get more intuition how equivariant kernels, not necessarily GIM-kernels, behave, we want to sketch how such kernels can be interpreted as estimates of the relative pose of two objects. First we consider the probably most simple equivariant kernel, the complex hermitian inner product itself. The group $U(1)$ consists of all complex numbers of unit length. If we define the representation of $U(1)$ acting on the $\mathbb{C}^n$ by a simple scalar multiplication with a complex unit number, then the ordinary inner product is obviously equivariant, that is

$$\langle g_1 \mathbf{x}_1 | g_2 \mathbf{x}_2 \rangle = \langle e^{\mathbf{i}\phi_1} \mathbf{x}_1 | e^{\mathbf{i}\phi_2} \mathbf{x}_2 \rangle = e^{\mathbf{i}\phi_1} \langle \mathbf{x}_1 | \mathbf{x}_2 \rangle e^{-\mathbf{i}\phi_2} = g_1 \langle \mathbf{x}_1 | \mathbf{x}_2 \rangle g_2^{\dagger}.$$

The absolute value of $\langle \mathbf{x}_1 | \mathbf{x}_2 \rangle$ can be interpreted as some kind of similarity. But how one should interpret the complex phase of $\langle \mathbf{x}_1 | \mathbf{x}_2 \rangle$? Therefore consider the distance $J(\phi) = ||\mathbf{x}_1 - e^{\mathbf{i}\phi}\mathbf{x}_2||$ for different values of ϕ. It is easy to show that J is minimal if ϕ is chosen according to $e^{\mathbf{i}\phi} = \frac{\langle \mathbf{x}_1 | \mathbf{x}_2 \rangle}{|\langle \mathbf{x}_1 | \mathbf{x}_2 \rangle|}$. Thus, the phase of the inner product carries information about the relative pose of the two objects. Actually this result can be generalized for equivariant kernels. Assume that we want to align the featuremaps $\Psi(\mathbf{x}_1)$ and $\Psi(\mathbf{x}_2)$ of an equivariant kernel optimally with respect to the group $\mathcal{G}$, that is, we try to minimize

$$J(g) = ||\Psi(\mathbf{x}_1) - \Psi(g\mathbf{x}_2)||^2,$$

Reformulating the objective leads to maximizing the trace

$$J'(g) = \mathrm{tr}(K(\mathbf{x}_1, \mathbf{x}_2)\rho_g^{\dagger} + \rho_g K(\mathbf{x}_2, \mathbf{x}_1)),$$

Let us assume that the group representation is surjective, that is, for any unitary matrix there is a $g \in \mathcal{G}$ such that ρ_g is identical to this matrix. Then we can use the Singular Value Decomposition (SVD) of the kernel to get the optimal solution. Let $K(\mathbf{x}_1, \mathbf{x}_2) = V\Sigma W$ the SVD, then one can show that $\rho_{g^*} = VW$ gives the optimal alignment, where J' takes $J'(g^*) = 2\mathrm{tr}(\Sigma)$ in the optimum. Interpreting this result, we can say that the sum of the singular values of a equivariant kernel expresses the similarity of the two given objects, while the unitary parts U and V contain information about the relative pose of the objects. And in fact, if $\mathbf{x}_1 = g'\mathbf{x}_2$, then $g^* = g'$ holds.

We want to demonstrate the proposed behavior with another example. Consider the finite group of cyclic translations C_n acting on the n dimensional vector space $\mathcal{X} = \mathbb{C}^n$. The group elements g_k with $k = 0, \ldots, n-1$ act on this space by $[\rho_{g_k}\mathbf{x}]_l := [\mathbf{x}]_{(l-k) \bmod n}$. This is just one particular representation of C_n. Another representation for example acts by scalar multiplications $\rho'_{g_k}\mathbf{y} := e^{\mathbf{i}\frac{2\pi}{n}k}\mathbf{y}$. Let us consider a linear GIM-kernel whose input space is $\mathcal{X} = \mathbb{C}^n$ with the representation ρ_g

and whose output space is $\mathcal{Y} = \mathbb{C}$ with the representation ρ'_g. One can choose $\mathcal{Y}$ one dimensional because ρ'_g just acts by scalar multiplications. The GIM-kernel $K : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{C}$ is given by

$$K(\mathbf{x}_1, \mathbf{x}_2) = \sum_{k=0}^{n-1} \langle \mathbf{x}_1 | \rho_{g_k} \mathbf{x}_2 \rangle \, e^{\mathbf{i}\frac{2\pi}{n}k}.$$

To interpret the behavior of K and understand with respect to which properties the objects are aligned we consider the feature $\Psi(\mathbf{x})$. Having a closer look the feature is nothing else than the first coefficient of a discrete Fourier expansion of $\mathbf{x}$. Thus, the kernel is just the product of the first coefficients of object $\mathbf{x}_1$ with the complex conjugate coefficient of $\mathbf{x}_2$. Consequently, the unitary part of the kernel, the complex phase, represents the relative phase of the first Fourier modes of the objects. The alignment with respect to those is a very common procedure in image processing applications to obtain complete invariant feature sets for the cyclic translation group C_n or other abelian groups like the two dimensional rotations $SO(2)$ (see, for example, Canterakis, 1986). The considerations get more intricate when non-abelian groups are considered. For 3D rotations this subject is discussed by Reisert and Burkhardt (2005) in more detail.

2.4 Equivariant Representer Theorem

In the beginning of this section we motivated our approach by pretending to present training patterns in all possible poses, where the pose of a target is implicitly turned by letting the group act on the regression coefficients. We did not clarify whether this is the optimal solution to obtain an equivariant behavior. Are there other possible solutions which are more general but also equivariant?

It is easy to check that the subspace $\mathcal{E} \subset \mathcal{Z}_k$ of equivariant functions is a linear subspace. The projection $\Pi_\mathcal{E}$ onto this subspace is given by

$$(\Pi_\mathcal{E}\mathbf{f})(\mathbf{x}) = \frac{1}{\mu(\mathcal{G})} \int_\mathcal{G} g^{-1}\mathbf{f}(g\mathbf{x}) \, dg. \tag{3}$$

This projection operator is the core of the construction principle presented in this work. In classical invariant theory this operator is also known as the Reynolds operator (Mumford et al., 1994).

Lemma 2.6 *The projection $\Pi_\mathcal{E}$ of (1) admits the proposed representation given in (2).*

Proof Applying the projection on (1) yields

$$\begin{aligned}(\Pi_\mathcal{E}\mathbf{f})(\mathbf{x}) &= \frac{1}{\mu(\mathcal{G})} \int_\mathcal{G} \sum_{i=1}^{n} k(g\mathbf{x}, \mathbf{x}_i) \, g^{-1}\mathbf{a}_i \, dg = \frac{1}{\mu(\mathcal{G})} \int_\mathcal{G} \sum_{i=1}^{n} k(\mathbf{x}, g^{-1}\mathbf{x}_i) \, g^{-1}\mathbf{a}_i \, dg \\ &= \frac{1}{\mu(\mathcal{G})} \int_\mathcal{G} \sum_{i=1}^{n} k(\mathbf{x}, h\mathbf{x}_i) h\mathbf{a}_i \, dh,\end{aligned}$$

where we used the substitution $h = g^{-1}$. ■

In fact, this projection is an unitary projection with respect to the naturally induced scalar product in the feature space. Before stating this more precisely we show how vector valued functions which are based on scalar kernels (as introduced in Eq. (1)) look in feature space. As already mentioned the evaluation functionals and the representations of the functions live in different spaces. The function of form (1) has the following representation $\mathbf{f}(\mathbf{x}) = \langle \Phi(\mathbf{x}) \otimes I_{\mathcal{Y}} | \Psi_{\mathbf{f}} \rangle_{\mathcal{H}}$. The matrix $I_{\mathcal{Y}} \in L(\mathcal{Y})$ denotes the identity matrix on $\mathcal{Y}$. The corresponding feature-space representation $\Psi_{\mathbf{f}}$ is an element of the contracted feature-space $\mathcal{F} \otimes \mathcal{Y}$, a $L(\mathcal{Y})$-left-module. This is easy to see when you think of $\mathcal{F} \otimes \mathcal{Y}$ as vectors whose components itself are column vectors. These column vectors only admit multiplication by matrices from the left as we know from ordinary matrix calculus. The feature space representation of the function is obviously of the form $\Psi_{\mathbf{f}} = \sum_i \Phi(\mathbf{x}_i) \otimes \mathbf{a}_i$, where the $\mathbf{a}_i$ are vector-valued expansion coefficients. The naturally induced scalar-valued inner product in this space is given by

$$\langle \Phi \otimes \mathbf{a} | \Phi' \otimes \mathbf{a}' \rangle_{\mathcal{F}\otimes\mathcal{Y}} = \langle \Phi | \Phi' \rangle_{\mathcal{F}} \langle \mathbf{a} | \mathbf{a}' \rangle_{\mathcal{Y}}. \tag{4}$$

The projection operator $\Pi_{\mathcal{E}}$ as defined in Eq. (3) acts directly on the functions $\mathbf{f}$. To show that $\Pi_{\mathcal{E}}$ is unitary we must examine how it acts on the feature space representation. Therefore, we define the group action on $\mathcal{F} \otimes \mathcal{Y}$

$$g(\Phi(\mathbf{x}) \otimes \mathbf{a}) := \Phi(g^{-1}\mathbf{x}) \otimes (g^{-1}\mathbf{a}). \tag{5}$$

By the use of this we define a new operator $\tilde{\Pi}_{\mathcal{E}}$ acting on $\mathcal{F} \otimes \mathcal{Y}$ and show that it corresponds to $\Pi_{\mathcal{E}}$ and that it is unitary with respect to the inner product given in Eq. (4).

Proposition 2.7 *If $\tilde{\Pi}_{\mathcal{E}}$ is given by*

$$\tilde{\Pi}_{\mathcal{E}} \Psi_{\mathbf{f}} := \frac{1}{\mu(\mathcal{G})} \int_{\mathcal{G}} g \Psi_{\mathbf{f}} \, dg$$

then $\tilde{\Pi}_{\mathcal{E}} \Psi_{\mathbf{f}} = \Psi_{\Pi_{\mathcal{E}}\mathbf{f}}$ holds and $\tilde{\Pi}_{\mathcal{E}}$ is a unitary projection.

Proof Let $\Psi_{\mathbf{f}} = \sum_i \Phi(\mathbf{x}_i) \otimes \mathbf{a}_i$ and correspondingly $f(\mathbf{x}) = \sum_{i=1}^n k(\mathbf{x}, \mathbf{x}_i)\mathbf{a}_i$. Then

$$\begin{aligned}
(\Pi_{\mathcal{E}}\mathbf{f})(\mathbf{x}) &= \frac{1}{\mu(\mathcal{G})} \int_{\mathcal{G}} \sum_{i=1}^n k(g\mathbf{x}, \mathbf{x}_i) \, g^{-1}\mathbf{a}_i \, dg = \frac{1}{\mu(\mathcal{G})} \int_{\mathcal{G}} \langle \Phi(g\mathbf{x}) \otimes \rho_g | \Psi_{\mathbf{f}} \rangle_{\mathcal{H}} \\
&= \langle \Phi(\mathbf{x}) \otimes I_{\mathcal{Y}} | \frac{1}{\mu(\mathcal{G})} \int_{\mathcal{G}} g \Psi_{\mathbf{f}} \rangle_{\mathcal{H}} = \langle \Phi(\mathbf{x}) \otimes I_{\mathcal{Y}} | \tilde{\Pi}_{\mathcal{E}} \Psi_{\mathbf{f}} \rangle_{\mathcal{H}}
\end{aligned}$$

this yields $\tilde{\Pi}_{\mathcal{E}} \Psi_{\mathbf{f}} = \Psi_{\Pi_{\mathcal{E}}\mathbf{f}}$. The operator $\tilde{\Pi}_{\mathcal{E}}$ is obviously a projection, because $\tilde{\Pi}_{\mathcal{E}}^2 = \tilde{\Pi}_{\mathcal{E}}$. The unitarity is due to unimodularity of the group. It is easily proved by showing that $\tilde{\Pi}_{\mathcal{E}}$ is hermitian with respect to the inner product in Eq. (4). ■

In conclusion, due to the unitartity the projection $\tilde{\Pi}_{\mathcal{E}}$ gives the best equivariant approximation of an arbitrary function in $\mathcal{Z}_k$. The space of equivariant function $\mathcal{E}$ is nothing else than the space of fix points with respect to the group action defined in Eq. (5), that is $\mathcal{E} = \{\Psi \in \mathcal{F} \otimes \mathcal{Y} | \forall g \in \mathcal{G} : g\Psi = \Psi\}$.

The proof of the following theorem is similar to the proof given in Schoelkopf and Smola (2002) for the general Representer Theorem. Note that we allow coupling between the training samples.

Theorem 2.8 (Equivariant Representer Theorem) *Let $\Omega : \mathbb{R}^+ \mapsto \mathbb{R}$ be a strictly monotonically increasing function and $c : (\mathcal{X} \times \mathcal{Y} \times \mathcal{Y})^n \mapsto \mathbb{R}$ a loss function. Then each equivariant minimizer $\mathbf{f} \in \mathcal{Z}_k$ of*

$$R(\mathbf{f}) = c(\mathbf{x}_1, \mathbf{y}_1, \mathbf{f}(\mathbf{x}_1), ..., \mathbf{x}_n, \mathbf{y}_n, \mathbf{f}(\mathbf{x}_n)) + \Omega(||\mathbf{f}||^2)$$

is of the form

$$\mathbf{f}(\mathbf{x}) = \int_{\mathcal{G}} \sum_{i=1}^{n} k(\mathbf{x}, g\mathbf{x}_i)\ g\mathbf{a}_i\ dg,$$

Proof We can decompose a function $\mathbf{f} \in \mathcal{Z}_k$ orthogonally (in the sense of the inner product given in (4)) into a part in the span of

$$\{\Phi(\mathbf{x}_i) \otimes \mathbf{y} \in \mathcal{F} \otimes \mathcal{Y} | i = 1..n, \mathbf{y} \in \mathcal{Y}\}$$

and its orthogonal complement;

$$\mathbf{f}(\mathbf{x}) \quad = \quad \mathbf{f}_{||}(\mathbf{x}) + \mathbf{f}_{\perp}(\mathbf{x}).$$

We know by construction that $\mathbf{f}_{\perp}(\mathbf{x}_i) = 0$ for all $i = 1..n$. Every equivariant function in $\mathcal{Z}_k$ is a projection of the form $\Pi_{\mathcal{E}}\mathbf{f}$. Considering the projection of the decomposition

$$(\Pi_{\mathcal{E}}\mathbf{f})(\mathbf{x}) \quad = \quad (\Pi_{\mathcal{E}}\mathbf{f}_{||})(\mathbf{x}) + (\Pi_{\mathcal{E}}\mathbf{f}_{\perp})(\mathbf{x})$$

we also know that the second term $(\Pi_{\mathcal{E}}\mathbf{f}_{\perp})(\mathbf{x}_i) = 0$ vanishes for all training samples. Thus the loss function in $R(\Pi_{\mathcal{E}}\mathbf{f})$ stays unchanged if one neglects $\Pi_{\mathcal{E}}\mathbf{f}_{\perp}$. By Lemma 2.6 we know that $\Pi_{\mathcal{E}}\mathbf{f}_{||}$ is of the proposed form

$$\Pi_{\mathcal{E}}\mathbf{f}_{||} = \int_{\mathcal{G}} \sum_{i=1}^{n} k(\mathbf{x}, g\mathbf{x}_i)\ g\mathbf{a}_i\ dg.$$

Due to the orthogonality we also know

$$||\Pi_{\mathcal{E}}\mathbf{f}||^2 \quad = \quad ||\Pi_{\mathcal{E}}\mathbf{f}_{||}||^2 + ||\Pi_{\mathcal{E}}\mathbf{f}_{\perp}||^2$$

and hence $\Omega(||\Pi_{\mathcal{E}}\mathbf{f}||^2) \geq \Omega(||\Pi_{\mathcal{E}}\mathbf{f}_{||}||^2)$. And thus for fixed values of $\mathbf{a}_i$ we get $R(\Pi_{\mathcal{E}}\mathbf{f}) \geq R(\Pi_{\mathcal{E}}\mathbf{f}_{||})$, so we have to choose $\Pi_{\mathcal{E}}\mathbf{f}_{\perp} = 0$ to minimize the objective. As this also has to hold for the solution, the theorem holds. ■

Note that the above theorem makes only statements for vector valued function spaces induced by a scalar kernel. There are also matrix valued kernels that are not based on a scalar kernels, that is, GIM-kernels are not the only way to obtain equivariant kernels. In Section 4 we will work out that one important class of equivariant kernels are GIM-kernels, namely those kernels whose corresponding group representation is irreducible.

2.5 Non-Compact Groups and Relation to Volterra Filters

In this work we consider compact unimodular groups. This is indeed very restrictive. Demanding that the functions $f(g) = k(\mathbf{x}_1, g\mathbf{x}_2)$ are all functions of compact support, it seems possible to generalize our theory for non-compact groups.

In signal-processing the group of time-shifts is elementary, which is actually a non-compact unimodular group. Convolutions are known to be linear time-invariant mappings. In our terms convolutions are equivariant to time-shifts. The so called Volterra series (Boyd et al., 1984) are generalized non-linear convolutions. The Volterra theory states that a nonlinear system, that maps a function $\mathbf{x}$ defined on the time line on a function $\mathbf{f}$ in an equivariant manner, can be modeled as infinite sums of multidimensional convolutions of increasing order

$$[\mathbf{f}(\mathbf{x})]_t = h_0 + \int_{g \in \mathcal{G}} h_1(g)[g\mathbf{x}]_t \, dg + \int_{g \in \mathcal{G}} \int_{g' \in \mathcal{G}} h_2(g,g')[g\mathbf{x}]_t[g'\mathbf{x}]_t \, dg dg' + ...,$$

where $[\mathbf{x}]_t$ denotes the value of $\mathbf{x}$ at time t and $\mathcal{G}$ is the group of time-shifts. In fact, Volterra series of degree n can be modeled with polynomial matrix kernels of degree n, that is, the scalar basis kernel is $k(\mathbf{x}_1,\mathbf{x}_2) = (1 + \langle \mathbf{x}_1|\mathbf{x}_2 \rangle)^n$. In other words, the RKHS of the induced matrix kernel is the space of n-th order Volterra series. Using the exponential kernel $k(\mathbf{x}_1,\mathbf{x}_2) = e^{\lambda \langle \mathbf{x}_1|\mathbf{x}_2 \rangle}$ the RKHS is actually the space of infinite degree Volterra series. For higher order polynomial kernels or even exponential kernels a kernelization is quite useful, because the run-time is linear in the number of training samples instead of proportional to the size of the induced feature space. There is already work by Dodd and Harrison (2003) which proposes a similar kernelized version of Volterra theory, but our theory is much more general. Our work tries to give a kernelized generalization of Volterra theory for arbitrary unimodular groups from a machine learning point of view.

3. Constructing Kernels

Similar to scalar valued kernels there are also several building rules to obtain new matrix and scalar valued kernels from existing ones. Most of them are similar to the scalar case, but there are also 'new' ones. In particular, the rule for building a kernel out of two kernels by multiplying them splits into two rules, either using the tensor product or the matrix product.

Proposition 3.1 (Closure Properties) *Let $K_1 : \mathcal{X} \times \mathcal{X} \mapsto L(\mathcal{Y})$ and $K_2 : \mathcal{X} \times \mathcal{X} \mapsto L(\mathcal{Y}')$ be matrix valued kernels and $A \in L(\mathcal{V},\mathcal{Y})$ with full row-rank, then the following functions are kernels.*

a) Let $\mathcal{Y} = \mathcal{Y}'$. $K(\mathbf{x}_1,\mathbf{x}_2) = K_1(\mathbf{x}_1,\mathbf{x}_2) + K_2(\mathbf{x}_1,\mathbf{x}_2)$

b) $K(\mathbf{x}_1,\mathbf{x}_2) = A^\dagger K_1(\mathbf{x}_1,\mathbf{x}_2)A$

c) $K(\mathbf{x}_1,\mathbf{x}_2) = K_1(\mathbf{x}_1,\mathbf{x}_2) \otimes K_2(\mathbf{x}_1,\mathbf{x}_2)$

d) Let $\mathcal{Y} = \mathcal{Y}'$. If for all $\mathbf{x} \in \mathcal{X}$, $K_1(\mathbf{x},\mathbf{x})$ and $K_2(\mathbf{x},\mathbf{x})$ commute, then the matrix product $K(\mathbf{x}_1,\mathbf{x}_2) = K_1(\mathbf{x}_1,\mathbf{x}_2)\ K_2(\mathbf{x}_1,\mathbf{x}_2)$ is a kernel.

e) $K(\mathbf{x}_1,\mathbf{x}_2) = K_1(\mathbf{x}_1,\mathbf{x}_2)^\dagger$

f) $k(\mathbf{x}_1,\mathbf{x}_2) = tr(K_1(\mathbf{x}_1,\mathbf{x}_2))$

Proof We omit proofs for a) and b) since the reasoning directly translates from the scalar case. To show c) for matrix kernels we give the corresponding feature-maps in terms of the already known feature-maps Ψ_1, Ψ_2 for kernel K_1 and K_2. From ordinary tensor calculus we know that

$$\langle \Psi_1 \otimes \Psi_2 | \Psi_1' \otimes \Psi_2' \rangle_{K_1 \otimes K_2} = \langle \Psi_1 | \Psi_1' \rangle_{K_1} \otimes \langle \Psi_2 | \Psi_2' \rangle_{K_2},$$

so the new feature-map for the tensor-product kernel c) is obviously $\Psi = \Psi_1 \otimes \Psi_2$. The positivity is also given, since the tensor product of two positive definite matrices is again positive definite. For d) the feature-map is actually the same as for c), but we define a different inner product by

$$\langle \Psi_1 \otimes \Psi_2 | \Psi_1' \otimes \Psi_2' \rangle_{K_1 K_2} := \langle \Psi_1 | \Psi_1' \rangle_{K_1} \langle \Psi_2 | \Psi_2' \rangle_{K_2},$$

which extends uniquely to the whole space and is well defined. Since we assume that the diagonal kernels $K_1(\mathbf{x},\mathbf{x})$ and $K_2(\mathbf{x},\mathbf{x})$ commute, the positive definiteness is also given, because the eigenvalues of the matrix product are just the products of the positive eigenvalues of its factors. The proof of e) is trivial. For the proof of statement f), we only have to recall Remark 2.4, where we mentioned that the trace of a positive matrix valued product is an ordinary positive scalar inner product.

■

Let us have a look how such rules can be applied to equivariant kernels and under what circumstances equivariance is preserved. Rule a) can be applied if K_1 and K_2 have the same transformation behavior, meaning that the underlying group representations are identical. Rule b) preserves equivariance if the matrix A is unitary. One can see this by redefining the group representation on $\mathcal{Y}$ by $\rho_g' = A^\dagger \rho_g A$. Since ρ_g' is again an unitary representation, K is an equivariant kernel. Rule c) can also be used to construct equivariant kernels. Supposing an equivariant kernel K_1 and an invariant scalar kernel k in sense that $k(\mathbf{x}_1, g\mathbf{x}_2) = k(\mathbf{x}_1, \mathbf{x}_2)$ for all $g \in \mathcal{G}$, then rule c) implies that $K = K_1 \otimes k = K_1 k$ is also a kernel and in fact equivariant. But how can we construct an invariant kernel k. The simplest approach is to use invariant features, but our framework also offers two possibilities. A GIM-kernel based on the trivial representation (just $\rho_g = 1$) is obviously invariant (this special case covers the approach proposed by Haasdonk et al., 2005). Another possibility is to use rule d) and f) to obtain an invariant kernel. But therefore we have to force the kernel to be normal, that is, the kernel K must always commute with its transpose $K^\dagger$.

Lemma 3.2 *If K is a normal equivariant kernel, then $k = tr(K^\dagger K)$ is an invariant kernel in the following sense $k(\mathbf{x}_1, g\mathbf{x}_2) = k(g'\mathbf{x}_1, \mathbf{x}_2) = k(\mathbf{x}_1, \mathbf{x}_2)$ for all $g, g' \in \mathcal{G}$.*

Proof Since K is normal $K(\mathbf{x},\mathbf{x})$ commutes with its transpose and hence $K^\dagger K$ is a kernel by rule d). By rule f) we know that $\mathrm{tr}(K^\dagger K)$ is a kernel. The invariance of the trace

$$k(\mathbf{x}_1, g\mathbf{x}_2) = \mathrm{tr}((K\rho_g^\dagger)^\dagger K \rho_g^\dagger) = \mathrm{tr}(\rho_g K^\dagger K \rho_g^\dagger) = \mathrm{tr}(K^\dagger K) = k(\mathbf{x}_1, \mathbf{x}_2)$$

proves invariance of the kernel. ■

4. Irreducible Kernels

The question arises which representations should be used to construct GIM-kernels. There are many possibilities, but we want to choose, of course, the most simple and computationally most efficient ones. A matrix kernel which is diagonal seems to be the most appropriate; this means that the kernel has to be diagonal for every pair of input patterns. If the kernel is diagonal the number of kernel entries that has to be computed is just the dimension of the output space $\mathcal{Y}$. But not every matrix kernel can be written in diagonal form, maybe there is only a block diagonal form or even only a fully occupied matrix. To formalize this more precisely we introduce the notion of *irreducibility* for matrix kernels, which should cover the intuition that a kernel cannot be decomposed in a more simple, more diagonal kernel.

Definition 4.1 *A kernel $K(\mathbf{x}_1,\mathbf{x}_2) \in L(\mathcal{Y})$ is reducible if there is a nonempty subspace $\mathcal{W} \subset \mathcal{Y}$, such that for every $\mathbf{y} \in \mathcal{W}$ and for every $\mathbf{x}_1,\mathbf{x}_2 \in X$, we have $K(\mathbf{x}_1,\mathbf{x}_2)\mathbf{y} \in \mathcal{W}$, otherwise the kernel is called irreducible.*

This definition is very close to the notion of *irreducibility* for group representations. This notion plays the central role in the harmonic analysis of groups (see Miller, 1991). A group representation $\rho_g \in L(\mathcal{V})$ is called reducible if there exists a nonempty subspace $\mathcal{W} \subset \mathcal{V}$ such that for all $\mathbf{w} \in \mathcal{W}$ also $\rho_g\mathbf{w} \in \mathcal{W}$. Then $\mathcal{W}$ is called a invariant subspace. If a representation is not reducible it is called irreducible. For all abelian groups the irreducible representations are very simple and all one-dimensional. They are just the complex numbers of unit length. Most of the readers may know them as Fourier transformations. For example, for the group of two dimensional rotation $SO(2)$, they are just $\rho_g^{(l)} = e^{\mathbf{i}l\phi}$. The invariant subspace on which $\rho_g^{(l)}$ is acting is just $\mathbb{C}$, a one dimensional vector space. Suppose a given function on the circle, then the lth component of the Fourier representation of this function is exactly the representation space on which the representation $\rho_g^{(l)}$ is acting on. For abelian groups the basis function on which the irreducible representation are working have the same formal appearance as the irreducible representations itself which might be confusing. For non-abelian groups it is different. Consider the group of three dimensional rotations $SO(3)$. The corresponding irreducible representations are the so called D-Wigner matrices which are not one dimensional anymore. They are matrices of increasing size $\rho_g^{(l)} \in \mathbb{C}^{(2l+1)\times(2l+1)}$. They act obviously on $2l+1$ dimensional vector spaces. For a given function defined on a three-dimensional sphere the expansion in the invariant subspaces is the so called Spherical Harmonic expansion which is also sometimes called the Fourier representation of the 2-Sphere. One may call harmonic analysis the generalization of Fourier theory to non-abelian groups.

If an equivariant kernel transforms by an irreducible representation, then it can be deduced that also the kernel itself is irreducible. But the opposite direction is not true in general, otherwise any equivariant kernel would be a GIM-kernel. We will later discuss this in more detail. But first two basic properties.

Lemma 4.2 *If the representation ρ of a strictly positive definite, equivariant kernel K is irreducible then K is also irreducible.*

Proof Assume that K is reducible, then there is a subspace $\mathcal{W} \subset \mathcal{Y}$, such that $\mathbf{y} \in \mathcal{W} \Rightarrow K(\mathbf{x}_1,\mathbf{x}_2)\mathbf{y} \in \mathcal{W}$. Since ρ is irreducible, we can choose $\mathbf{y} \in \mathcal{W}$ such that there exists a g with $\rho_g\mathbf{y} = \mathbf{w} + \mathbf{w}^\perp \in \mathcal{W} \oplus \mathcal{W}^\perp$. But we also know that $K(\mathbf{x}_1, g^{-1}\mathbf{x}_2)\mathbf{y} = K(\mathbf{x}_1,\mathbf{x}_2)\rho_g\mathbf{y} \in \mathcal{W}$. In particular, we can choose $\mathbf{x}_1 = \mathbf{x}_2 = \mathbf{x}$. Due to the reducibility of K and the regularity (because of strictly pd) of $K(\mathbf{x},\mathbf{x})$, we know that $0 \neq K(\mathbf{x},\mathbf{x})\mathbf{w}^\perp \in \mathcal{W}^\perp$ and hence $K(\mathbf{x},\mathbf{x})\rho_g\mathbf{y} \notin \mathcal{W}$. Contradiction! ■

Since we know from representation theory (Gaal, 1973) that any unitary representation can be decomposed in a direct sum of irreducible representations we can make a similar statement for kernels.

Corollary 4.3 *Any GIM-kernel K can be decomposed in a direct sum of irreducible GIM-kernels associated with its irreducible representations.*

Proof Let ρ be the representation of the GIM-kernel. Since any ρ can be decomposed in a direct sum $\rho = \rho^{(1)} \oplus ... \oplus \rho^{(n)}$, we directly see that

$$K = K_1 \oplus .. \oplus K_n,$$

where $K_l(\mathbf{x}_1, \mathbf{x}_2) = \int_{\mathcal{G}} k(\mathbf{x}_1, g\mathbf{x}_2)\rho_g^{(l)} dg$. And the K_l are irreducible due to Lemma 4.2. ■

By the Peter-Weyl-Theorem (Gaal, 1973) we know that the entries of the irreducible representations of a compact group $\mathcal{G}$ form a basis for the space of square-integrable function on $\mathcal{G}$. We also know that the entries of the representations are orthogonal with respect to the canonical dot-product. The following Lemma is based on that.

Lemma 4.4 *If the representation ρ of an equivariant kernel K is irreducible then K is a GIM-kernel.*

Proof We define the corresponding scalar kernel by

$$k = \frac{n}{\mu(\mathcal{G})} \mathrm{tr}(K),$$

where n is the dimensionality of the associated group representation. Then

$$\int_{\mathcal{G}} k(\mathbf{x}_1, g\mathbf{x}_2)\rho_g dg = \frac{n}{\mu(\mathcal{G})} \int_{\mathcal{G}} \mathrm{tr}(K(\mathbf{x}_1, \mathbf{x}_2)\rho_g^\dagger)\rho_g dg.$$

By the orthogonality relations for irreducible group representations we know that

$$\frac{n}{\mu(\mathcal{G})} \int_{\mathcal{G}} \mathrm{tr}(K(\mathbf{x}_1, \mathbf{x}_2)\rho_g^\dagger)\rho_g dg = K(\mathbf{x}_1, \mathbf{x}_2).$$

■

It seems that we should concentrate on the irreducible representations of the considered group, and this is indeed the canonical way. Any scalar kernel can be written in terms of its irreducible GIM-kernel expansion.

Proposition 4.5 *Let k be a scalar kernel, then*

$$k(\mathbf{x}_1, \mathbf{x}_2) = tr(K(\mathbf{x}_1, \mathbf{x}_2)),$$

where $K = K_1 \oplus K_2 \oplus ...$ is the direct sum of its irreducible GIM-kernels as given in Corollary 4.3

Proof Due to the Peter-Weyl-Theorem we can expand the scalar kernel in terms of the irreducible representation of $\mathcal{G}$, where the expansion coefficients are by definition the corresponding GIM-kernels.

$$k(\mathbf{x}_1, g\mathbf{x}_2) = \sum_{l=0}^{\infty} \mathrm{tr}(K_l(\mathbf{x}_1, \mathbf{x}_2)(\rho_g^{(l)})^\dagger)$$

where $K_l(\mathbf{x}_1,\mathbf{x}_2) = \int_G k(\mathbf{x}_1, g\mathbf{x}_2)\rho_g^{(l)} dg$. And hence we have

$$k(\mathbf{x}_1, g\mathbf{x}_2) = \sum_{l=0}^{\infty} \mathrm{tr}(K_l(\mathbf{x}_1, g\mathbf{x}_2)) = \mathrm{tr}(K_1(\mathbf{x}_1, g\mathbf{x}_2) \oplus K_2(\mathbf{x}_1, g\mathbf{x}_2) \oplus \ldots) = \mathrm{tr}(K(\mathbf{x}_1, g\mathbf{x}_2)),$$

which proves the statement. ■

Hence we have a one-to-one correspondence between the scalar basis kernel k and the GIM-kernels K_l formed by the irreducible group representations $\rho_g^{(l)}$. So we have for GIM-kernels always the duality between the matrix- and scalar-valued kernels. For Non-GIM-kernels this one-to-one correspondence does not hold.

4.1 Non GIM-kernels

There is another very simple way to construct equivariant kernels. For example assume that $\mathcal{X} = \mathcal{Y}$ then $K(\mathbf{x}_1,\mathbf{x}_2) = |\mathbf{x}_1\rangle\langle\mathbf{x}_2|$ is obviously an equivariant kernel. This kernel may be seen as the matrix-valued analogon to the linear scalar-valued kernel $k(\mathbf{x}_1,\mathbf{x}_2) = \langle\mathbf{x}_2|\mathbf{x}_1\rangle_{\mathcal{X}}$ and indeed $k = \mathrm{tr}(K)$ holds. Note that it is in general not possible to reconstruct K from $\mathrm{tr}(K)$, this is only possible for GIM-kernels by Proposition 4.5. Such a non GIM-kernel can be easily obtained by the construction principle from above. But how to check whether a kernel is a GIM-kernel or not?

Corollary 4.6 *Let K be an irreducible equivariant kernel and let its corresponding representation be reducible, then K is not a GIM-kernel.*

Proof Assume K is of GIM-type then by Corollary 4.3 it is reducible. Contradiction! ■

This statement is the counterpart to Lemma 4.4, which says that if the group-action is irreducible we know that we have a GIM-kernel.

The simple kernel $K(\mathbf{x}_1,\mathbf{x}_2) = |\mathbf{x}_1\rangle\langle\mathbf{x}_2|$ is not of practical value, because regression functions which are built out of it are always proportional their input. So one needs clever feature-maps to get useful kernels.

4.2 Kernel Response for Symmetric Patterns

GIM-kernels provide an intrinsic mechanism to cope with symmetric patterns. A pattern $\mathbf{x} \in \mathcal{X}$ is called $\mathcal{U}$-symmetric if there is a subgroup $\mathcal{U}$ of $\mathcal{G}$ such that for all $u \in \mathcal{U}$ we have $u\mathbf{x} = \mathbf{x}$. To characterize the kernel response for such patterns we define a linear unitary projection on the space of $\mathcal{U}$-symmetric patterns as follows

$$\pi_{\mathcal{U}}\mathbf{x} := \frac{1}{\mu(\mathcal{U})} \int_{\mathcal{U}} g\mathbf{x}\, dg. \tag{6}$$

The operator $\pi_{\mathcal{U}}$ is a unitary projection due to the same arguments as used in the proof of Proposition 2.7. Using this operator we can state the following

Lemma 4.7 *Let K be a GIM-kernel. If $\mathbf{x}_1$ is $\mathcal{U}_1$-symmetric and $\mathbf{x}_2$ is $\mathcal{U}_2$-symmetric then*

$$K(\mathbf{x}_1,\mathbf{x}_2) = \pi_{\mathcal{U}_1} K(\mathbf{x}_1,\mathbf{x}_2) \pi_{\mathcal{U}_2}$$

holds.

Proof Let $\mathcal{G}/\mathcal{U}$ the quotient group. If $\mathbf{x}_2$ is $\mathcal{U}_2$-symmetric then

$$\begin{aligned} K(\mathbf{x}_1,\mathbf{x}_2) &= \frac{1}{\mu(\mathcal{G})}\int\limits_{\mathcal{G}} k(\mathbf{x}_1,g\mathbf{x}_2)\rho_g\,dg = \frac{1}{\mu(\mathcal{G})}\int\limits_{\mathcal{G}/\mathcal{U}_2}\int\limits_{\mathcal{U}_2} k(\mathbf{x}_1,gu\mathbf{x}_2)\rho_{gu}\,du\,dg \\ &= \frac{1}{\mu(\mathcal{G}/\mathcal{U}_2)}\int\limits_{\mathcal{G}/\mathcal{U}_2} k(\mathbf{x}_1,g\mathbf{x}_2)\,dg\,\underbrace{\left(\frac{1}{\mu(\mathcal{U}_2)}\int_{\mathcal{U}_2}\rho_u\,du\right)}_{\pi_{\mathcal{U}_2}}. \end{aligned}$$

As $\pi_{\mathcal{U}}$ is a projection and thus $\pi_{\mathcal{U}}^2 = \pi_{\mathcal{U}}$ we can insert a $\pi_{\mathcal{U}_2}$ in the last line from above for free and reverse the whole computation an obtain $K(\mathbf{x}_1,\mathbf{x}_2) = K(\mathbf{x}_1,\mathbf{x}_2)\pi_{\mathcal{U}_2}$. If $\mathbf{x}_1$ is $\mathcal{U}_1$-symmetric the reasoning is very similar. Due to the unimodularity of the group and its subgroups it is no problem to exchange g and u in the last expression from the first line. Hence, we can use the same arguments as for the $\pi_{\mathcal{U}_2}$-symmetry. ■

Assume now we have a trained function $\mathbf{f}(\mathbf{x})$ composed as a linear combination of GIM-kernels. What is the response of $\mathbf{f}(\mathbf{x})$ if $\mathbf{x}$ is $\mathcal{U}$-symmetric. Due to Lemma 4.7 it is obvious that $\mathbf{f}(\mathbf{x})$ is also $\mathcal{U}$-symmetric and the symmetrization is obtained via the operator defined in Eq. (6). This behavior is reasonable, because there is no reason why the output should violate the symmetry if the input does not spend information beyond. For example, if the output space $\mathcal{Y}$ is a space of probability distributions the idea of symmetrization via $\pi_{\mathcal{U}}$ is a good idea. Because, if we are uncertain about the original pose of an input object due to symmetry, we should vote for all possible outputs allowed by the symmetry in a additive manner. And the operator $\pi_{\mathcal{U}}$ actually works this way.

Now let us consider the opposite view. Suppose we want to build a function that maps a $\mathcal{U}$-symmetric input $\mathbf{x}_0$ to a not symmetric output $\mathbf{y}_0$, which is obviously an ill-posed problem. How does the function behave? We are not able to make a general statement, because it depends on the learning algorithm applied. But to get an intuition consider the function $\mathbf{f}(\mathbf{x}) = K(\mathbf{x},\mathbf{x}_0)\mathbf{y}_0$, which essentially behaves as desired, if $K(\mathbf{x},\mathbf{x}) \propto I_{\mathcal{Y}}$. By Lemma 4.7 we know that $\mathbf{f}(\mathbf{x}) = K(\mathbf{x},\mathbf{x}_0)(\pi_{\mathcal{U}}\mathbf{y}_0)$ holds, thus the function behaves as if the desired output was symmetrized beforehand.

5. Applications

We show two application examples that basically can be described as image processing applications. We present a rotation-equivariant filter, which can be used for example, for image restoration or image enhancement. Thereby each pixel neighborhood is handled as one training instance, which leads to an enormous number of training samples. Hence, we let the filter work in the featurespace of a second-order matrix kernel. Secondly, we build a rotation equivariant object detector. Specific interest points in the image are detected. Relative to local features around these interest points the center of the object is learned. Of course ambiguities can occur. For example, relative to a corner of a rectangle there are two possible centers of the rectangle (only for a square there is a unique center). To cope with such ambiguities we learn the probability distribution that the center of the object lies in a specific direction relative to the local features around the interest points.

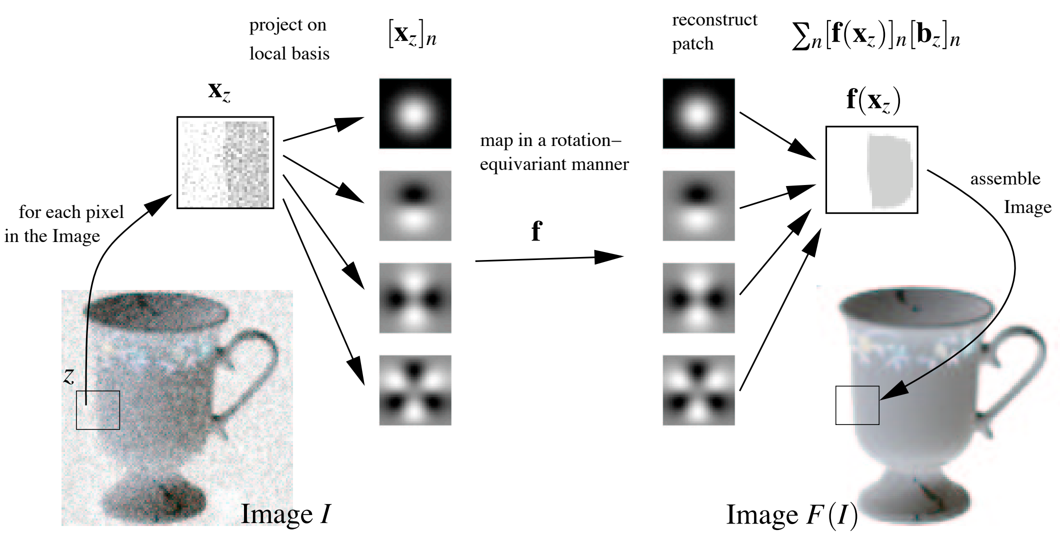


Figure 1: The workflow of the rotation equivariant filter for the example of image denoising.

5.1 Equivariant Kernels for 2D Rotations

In both experiments we want to learn functions $\mathbf{f} : \mathcal{X} \mapsto \mathcal{X}$, where $\mathcal{X}$ is the space of functions defined on the unit circle. The irreducible representations of 2D-rotations are $e^{\mathbf{i}n\phi}$ and the irreducible subspaces correspond to the Fourier representation of the function. Hence in Fourier representation the GIM-kernel is a diagonal kernel with entries

$$K_n(\mathbf{x}_1, \mathbf{x}_2) = \int_0^{2\pi} k(\mathbf{x}_1, g_\phi \mathbf{x}_2) e^{\mathbf{i}n\phi} \, d\phi \tag{7}$$

on the diagonal. Discrete approximations of this integral can be computed quickly by a Fast Fourier Transform (FFT). Assuming k is a dot-product kernel $k(\langle \mathbf{x}_1 | \mathbf{x}_2 \rangle_{\mathcal{X}})$, then a kernel evaluation looks as follows: Compute the cross-correlation $c(\phi) = \langle \mathbf{x}_1 | g_\phi \mathbf{x}_2 \rangle_{\mathcal{X}}$ using the FFT, apply the nonlinearity $k(c(\phi))$ and transform it back into Fourier domain $K_n = \int_0^{2\pi} k(c(\phi)) e^{\mathbf{i}n\phi} \, d\phi$. If the patterns are already in Fourier domain, we need two FFT per kernel evaluation. This approach is used in the second experiment.

The probably most simple non-linear scalar kernel is $k(\mathbf{x}_1, \mathbf{x}_2) = \langle \mathbf{x}_1 | \mathbf{x}_2 \rangle_{\mathcal{X}}^2$. Using this kernel as a scalar base kernel the matrix-valued feature space looks very simple. The feature map is given by

$$[\Psi(\mathbf{x})]_{nl} \propto [\mathbf{x}]_{-l} \, [\mathbf{x}]_{l+n}, \tag{8}$$

where $[\mathbf{x}]_l$ denotes lth component of the Fourier representation of $\mathbf{x}$. The induced matrix valued bilinear product in this space is the ordinary

$$K_n(\mathbf{x}_1, \mathbf{x}_2) = \sum_l [\Psi(\mathbf{x}_1)]_{nl} \, [\Psi(\mathbf{x}_2)]_{nl}^*.$$

hermitian inner product. We use a similar feature space representation in the first experiment.

5.2 A Rotation Equivariant Filter for Images

For convenience we represent a 2D gray-value image I in the complex plane, where we denote the gray value at point $z \in \mathbb{C}$ by I_z. The idea is to learn a rotation-equivariant function $\mathbf{f} : \mathcal{X} \mapsto \mathcal{X}$, which maps each neighborhood $\mathbf{x}_z \in \mathcal{X}$ of a pixel z onto a new neighborhood $\mathbf{f}(\mathbf{x}_z) \in \mathcal{X}$ with properties given by learning samples. The neighborhood is represented by projections on local basis functions $[\mathbf{b}_z]_n = z^n e^{-\lambda|z|^2}$ for $n = 0, \ldots, N-1$. This basis is actually a orthogonal basis with respect to the standard inner product. In Figure 1 the appearance of the basis is depicted. In particular, for each pixel z' the inner product

$$[\mathbf{x}_{z'}]_n = \langle I_{z-z'} | z^n e^{-\lambda|z|^2} \rangle = \sum_{z \in [1,256]^2} I_{z-z'} z^n e^{-\lambda|z|^2}$$

of the image with the basis images is computed. As the basis images are only considerable different from zero in a small neighborhood around the origin, the N expansion coefficients $[\mathbf{x}_{z'}]_n$ represent the appearance of neighborhood of a pixel z. Actually the computation of the local basis expansion is nothing else than a cross correlation of the image with the basis images, which can be expressed as a convolution:

$$[\mathbf{x}_z]_n = I_z * [\mathbf{b}_{-z}]_n^\dagger.$$

Here $*$ denotes the convolution operator. In Figure 1 the approach is depicted for an image denoising application. The basis functions from above have the advantage that their transformation behavior is the same as for functions defined on a circle, that is, $[\mathbf{x}_z]_n \mapsto [\mathbf{x}_z]_n e^{\mathbf{i}n\phi}$ for rotations around z. The vector-components $[\mathbf{f}(\mathbf{x}_z)]_n$ are also interpreted as expansion coefficients of the neighborhood in this basis. Since the neighborhoods overlap we formulate one global function which maps the whole image I on a new image

$$F(I_z) \quad = \quad \sum_n [\mathbf{f}(\mathbf{x}_z)]_n * [\mathbf{b}_z]_n = \mathbf{f}(\mathbf{x}_z) \underline{*} \mathbf{b}_z.$$

This function is translation- and rotation-equivariant, since $\mathbf{f}$ is rotation-equivariant and the convolution is naturally translation-equivariant. The scalar basis kernel k for the GIM-kernel expansion of $\mathbf{f}$ is chosen by

$$k(\mathbf{x}_z, \mathbf{x}'_z) = (1 + \langle \mathbf{x}_z | \mathbf{x}'_z \rangle_{\mathcal{X}})^2.$$

The corresponding feature-map is nearly the same as in (8). Given two images I^{src} and I^{dest} our goal is to minimize the cost

$$\sum_{z \in [1,256]^2} |I_z^{\text{dest}} - F_{\mathbf{w}}(I_z^{\text{src}})|^2 + \gamma ||\mathbf{w}||^2,$$

where $\mathbf{w}$ is the parameter vector yielding $\mathbf{f}(\mathbf{x}) = \langle \Psi(\mathbf{x}) | \mathbf{w} \rangle_{\mathcal{H}}$, where $\mathcal{H}$ is the feature space of the above kernel. Hence $F_{\mathbf{w}}$ is linear in $\mathbf{w}$ since the convolution is a linear operation. The parameter γ is a regularization parameter. The actual training procedure is a simple ridge regression scheme.

In our experiments we use images of size 256^2. Note that each pixel-neighborhood is regarded as one training sample. Due to the overlap of the neighborhoods the cost function couples training samples. The grayvalues of the images are scaled to $[0,1]$. The neighborhood of a pixel is represented by $N = 8$ coefficients, where the width of the Gaussian is around 10 pixels. The corresponding feature-space is of dimension $N(N+1)/2 + N = 44$. As already mentioned the feature-space dimension is much less than the number of training-samples (256^2) and hence working in feature-space is recommended.

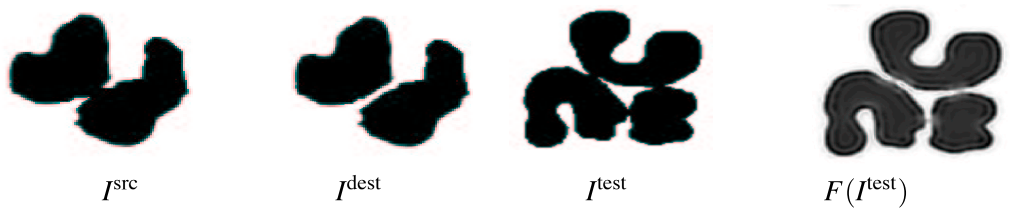


Figure 2: Results for the image transformation example. Images I^{src} and I^{dest} where used for training. Image I^{test} show the image for testing, the three connected should be separated. $F(I^{\text{src}})$ shows the image after application of the trained image transformation.

We conduct two experiments. First we train the filter to separate connected clusters of pixels. The left two images in Figure 2 show the images $I^{\text{src}}, I^{\text{dest}}$ used for training. As the equation system is highly overdetermined the regularization parameter γ can be chosen very small. If the training images are distorted by a small amount of independent Gaussian noise we can actually choose $\gamma = 0$. In general we can conclude that the higher the regularization γ is the less effective is our filter, but the more robust the filter is against additive noise. This is a well known behavior.

The right two images show a test image in its original appearance and after the application of the trained filter. One can see that the filter actually works properly. Of course, some artifacts are created. One can nicely see the equivariance of the filter. The behavior depends neither on the orientation of the touching areas nor the absolute position. In fact, the filter may be interpreted as a locally applied quadratic (because of the second-order polynomial kernel) Volterra filter. The test image I^{test} was chosen such that common methods would not work properly. For example, if one use a distance transform, the agglomerate would be divided in five instead three clusters, because the distance transform gives five local maxima.

As a second experiment we trained the filter to perform a deconvolution. For training we used a black disk on white background as target I^{dest} and a blurred version as I^{src} (blurred with a Gaussian of width 2 percent of the image size). In Figure 3 a) we show a blurred image which has to be sharpened (blurred with the same width as used for training). We compare our filter with an ordinary Wiener filter for deconvolution (see, for example, Jain, 1989). Assuming decorrelated Gaussian noise models, the Wiener filter only depends on the signal-to-noise ratio, which controls the intensity of the sharpening of the image. In our approach the regularization parameter γ plays a similar role as the signal-to-noise ratio for the Wiener filter. We tuned the parameters such that the obtained sharpness for both approaches are visually comparable. In Figure 3 b) we give the result obtained by our method, in c) the result for the Wiener filter. Our method produces a more natural looking image and more importantly it produces no over- and undershoots at the sharp edges like the Wiener filter.

5.3 A Rotation Equivariant Object Detector

The goal of the object detector is to find a specific object in an image independently of its rotational or translational pose. The proposed object detector may be seen as some kind of generalized hough transform (see Ballard, 1981), where the lookup-table is is replaced by an equivariant function,

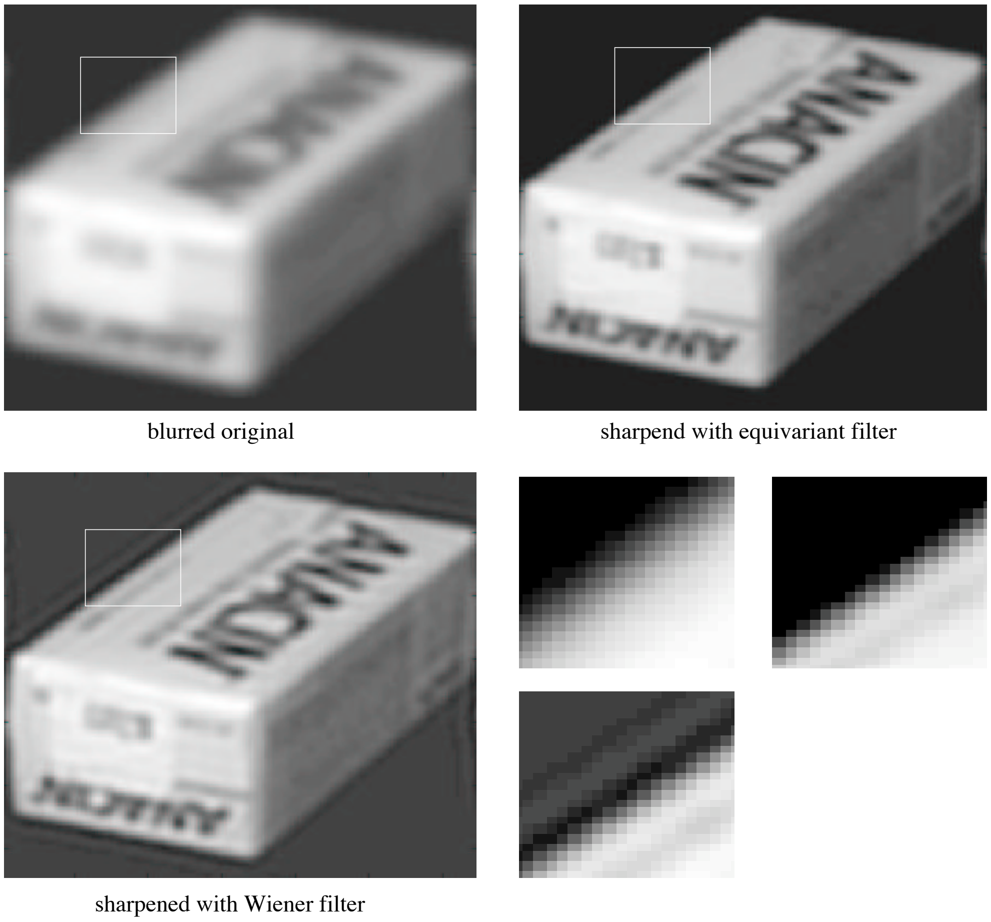

blurred original
sharpend with equivariant filter
sharpened with Wiener filter

Figure 3: Results for the image restoration example. The Wiener filter produces overshoot- and undershoot-artifacts; this is a well known problem for inverse linear filters. Our non-linear filter creates a fine and accurate edge while having the same sharpness as the Wiener filter. (have a look at these results in an electronic form, a printer may destroy the fine differences)

which is learned from a shape template. It has also some similarities to the object recognition system using the so called SIFT features (Lowe, 2004).

At first, stable interest points in the image have to be detected, for example, points of high variation or something similar. Relative to these interest points the algorithm votes for a predefined center of the object. Assume we want to detect rectangles in an image. Corners are an important and stable feature of a rectangle. Relative to a corner of a rectangle we have obviously two hypotheses for the putatively rectangle center. To cope with such multiple hypotheses in general we do not make discrete votes but vote for all possible points with its corresponding probability that there may be the center of the object. Our goal is to learn a mapping that maps features from the local neighborhood of the interest points to a probability distribution for the object's center. To detect the center of the object independently of its rotational pose this mapping has to be rotation-equivariant.

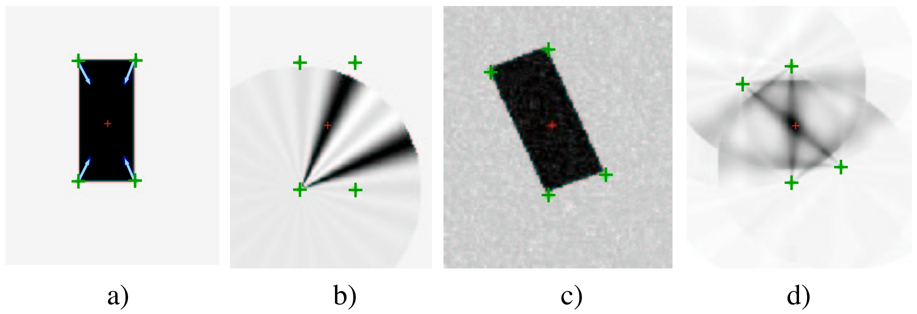


Figure 4: Example for the rectangle. a) The rectangle with its four interest points and its center. b) The response of the trained voting function $\mathbf{p}(\mathbf{x})$ for the lower left interest point (as mentioned the distance to the center is not learned, only the direction). The function votes for the center of the shown rectangle and for the center of a putatively rotated rectangle by 90 degrees. c) an rotated version of the training object with additive Gaussian noise. d) the superimposed responses of all four interest points. The maxima is obviously in the center of the rectangle.

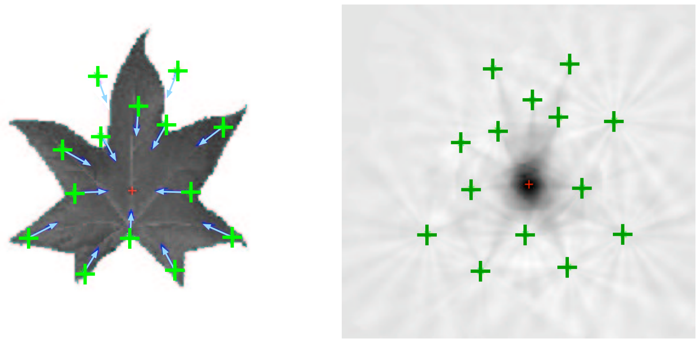
For simplicity we want to restrict ourselves to vote for points in the same specific direction with the same probability or weight, that is, the output of the equivariant-mapping is a probability distribution defined on the circle.

For the features we use the same projections on local basis-functions as in the previous experiment. To find the interest points we search for local maxima of the absolute value $|[\mathbf{x}_z]_2|$, which basically gives responses for corner-like structures. To eliminate responses at edges, which are rather unstable, we only take those local maxima whose values for the quotient $|[\mathbf{x}_z]_2|/|[\mathbf{x}_z]_1|$ are above a given threshold. Let $\mathbf{x}$ be a feature vector around some interest point and ϕ the angle to the center of the object. We are interested to learn the conditional distribution $p(\phi|\mathbf{x})$ from training samples $(\phi_i, \mathbf{x}_i)$ belonging to the training object's interest points. As we want to detect the objects in a rotation invariant manner, the distribution has to fulfill rotation equivariance $p(\phi|g_\varphi \mathbf{x}) = p(\phi + \varphi|\mathbf{x})$. In Section 4 we worked out that it is advantageous to work with irreducible kernels. As already mentioned the irreducible kernels of the 2D-rotation group act on the Fourier representation of functions, so we denote $p(\phi|\mathbf{x})$ by the vector $\mathbf{p}(\mathbf{x})$, where its components are the Fourier coefficients of $p(\phi|\mathbf{x})$ in respect to ϕ. We denote the cutoff-frequency of the Fourier representation by N.

We do not rigorously model $\mathbf{p}$ as a probability distribution. Let $\mathbf{e}_\phi$ be the unit-pulse in Fourier representation, then we minimize

$$\sum_i ||\mathbf{e}_{\phi_i} - \mathbf{p}(\mathbf{x}_i)||^2,$$

which is nothing else then the ordinary least-square regression scheme. But the solution basically behaves as we want it to. To get an idea, consider again the rectangle example with its four corners as interest points. If we neglect noise the local features $\mathbf{x}_i$ of the corners are all the same $\mathbf{x}_i = g_{\varphi_i}\mathbf{x}_0$ up to rotations g_{φ_i} of $0, 90, 180, 270$ degree. Then the equivariant least-square minimizer $\mathbf{p}(\mathbf{x}_0)$ is proportional to the Fourier transformed histogram of the relative angles $\phi_i + \varphi_i$. The reader should have an antipicatory look at Figure 4 a) and b), where this is illustrated. One can see that $\mathbf{p}(\mathbf{x}_0)$ give contributions for both possible corner directions.



a) b)

Figure 5: Training image a) The only training leaf. The interest points for training are marked and its corresponding direction to the center. b) The voting map for the training object. Black means high probability for the center, white low probability.

The scalar basis kernel is chosen to be the Gaussian kernel $k(\mathbf{x}_1, \mathbf{x}_2) = e^{-\lambda||\mathbf{x}_1 - \mathbf{x}_2||^2}$. The diagonal matrix kernel is approximated by using the FFT as depicted in Section 5.1. To get good approximations we have to perform frequency padding. In the experiments we used a FFT of size $6N$ (this is enough to compute a third-order polynomial kernel accurately). For accurate computation of the exponential kernel an infinite size FT would be needed, but the experiments have shown that an approximation is enough. The final learning procedure is very simple. We just have to solve N linear equations of the form

$$\mathbf{K}_n \mathbf{a}_n = \mathbf{b}_n \qquad n = 0, ..., N-1,$$

where the matrix entries of $(\mathbf{K}_n)_{i,j} = K_n(\mathbf{x}_i, \mathbf{x}_j)$ are the approximations of Equation (7), and the entries of the target are $[\mathbf{b}_n]_i = e^{\mathbf{i}n\phi_i}$.

In Figure 4 we illustrate results for the rectangle example. In a) the training example with its four interest points is shown and b) shows the contribution of the voting function of the lower left corner. The sinusoidal artifacts are stemming from the finite Fourier representation. In c) and d) we applied our object detector to a rotated and noisy rectangle. The results are satisfying, the voting function $\mathbf{p}(\mathbf{x})$ obviously behaves in an equivariant manner and is robust against small distortions. To make a more realistic example, we train our object detector to find leaves on natural background. In Figure 5 a) a segmented leaf for training is shown. We found by hand that 14 interest points are relatively stable. Of course, this number, depends on the size of the Gaussian used for feature computation (in our case the width is four percent of the image size, that is, rather large). After training we applied the object detector on the train image itself, which is shown in Figure 5 b). For each interest point we superimpose its contributions within a distance of about forty percent of the image size. To get equal contribution from each point we normalize the output of our voting function $\mathbf{p}(\mathbf{x})/||\mathbf{p}(\mathbf{x})||$. To show the stability and generalization ability of our object detector, we applied it on a more complex image shown in Figure 6 a). The leaf is from the same species as the training leaf but obviously different. The resulting voting map is shown in Figure 6 b), in fact, the overall maximum is in the center of the leaf, but it seems that the maximum is relatively unstable depending on the background. To improve the result, we modified the kernel. As explained in Section 3 we can use Lemma 3.2 to compute an invariant kernel and modify our matrix kernel by

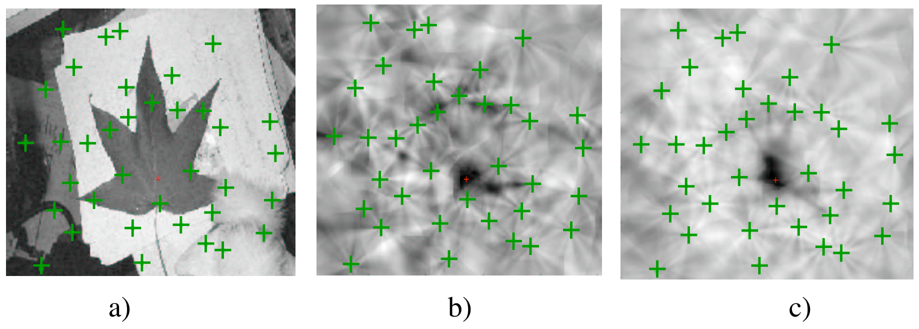


Figure 6: A test image of a leaf with background and small clutter. a) The testimage with the found interest points. b) The voting map for test image with the simple matrix kernel. c) The voting map achieved with the enhanced kernel for better selectivity.

$K' = K \operatorname{tr}(K^{\dagger}K)$ to give it more selectivity to the features itself. In Figure 6 c) the voting map with this enhanced kernel is shown, the maximum seems to be much more robust.

6. Conclusion and Outlook

We presented a new type of matrix valued kernel for learning equivariant functions. Several properties were shown and connections to representation theory were established. We showed with two illustrative examples that the theory is applicable.

The usage in nonlinear signal and image processing are apparent. Problems like image denoising, enhancement and image morphing can be tackled. Another simple task for the equivariant filter would be to gaps in road networks.

The proposed object detector shows in spite of its simplicity a nice behavior and generalization ability and should be worth to improve. Generalizations of our proposed experiments to 3D-rotations are straightforward.

From a theoretical point an extension of our framework to non-compact groups would be satisfying. Another important challenge is to find sparse learning algorithms with an unitary invariant loss functional. Unfortunately, the unitary extension of the ε-insensitive loss is not solvable via quadratic programming anymore. A last important issue is the local nature of transformations. In hand-written digit recognition invariance under rotations might turn a ”6” into a ”9”, while rotations by 5 degrees are ok. A generalization of our theory using notions like approximate or partial equivariance are necessary.

Acknowledgments

The authors would like to acknowledge the anonymous reviewers and the action editor for their useful suggestions and comments that have greatly improved the presentation of the paper. Further we would like to thank Nikos Canterakis and Bernard Haasdonk for carefully reading the manuscript in an early stage. And finally, thanks to Olaf Ronneberger for the fruitful discussions and inspiration.

References

L. Amodei. Reproducing kernels of vector-valued function spaces. In *Surface Fitting and Multiresolution Methods, A. Le Maut C. Rabut and L. L. Schumaker (eds.)*, pages 17–26, 1996.

N. Aronszajn. Theory of reproducing kernels. *Trans. AMS*, 686:337404, 1950.

D.H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13-2, 1981.

S. Boyd, L. O. Chua, and C. A. Desoer. Analytical foundations of Volterra series. *IMA Journal of Mathematical Control and Information*, 1:243–282, 1984.

J. Burbea and P. Masani. Banach and Hilbert spaces of vector-valued functions. *Pitman Research Notes in Mathematics*, 90, 1984.

H. Burkhardt and S. Siggelkow. Invariant features in pattern recognition - fundamentals and applications. In *In Nonlinear Model-Based Image/Video Processing and Analysis*, pages 269–307. John Wiley and Sons, 2001.

N. Canterakis. Vollstaendige minimale systeme von polynominvarianten fuer die zyklischen gruppen g(n) und fuer g(n) x g(n). In *In Tagungsband des 9. Kolloquiums - DFG Schwerpunkt: Digitale Signalverarbeitung*, pages 13–17, 1986.

T.J. Dodd and R.F. Harrison. Estimating Volterra filters in Hilbert space. In *Proceedings of IFAC Conference on Intelligent Control Systems and Signal Processing*, 2003.

A. Gaal. *Linear Analysis and Represenatation Theory*. Springer Verlag, New York, 1973.

B. Haasdonk, A. Vossen, and H. Burkhardt. Invariance in kernel methods by haar-integration kernels. In *Proceedings of the 14th Scandinavian Conference on Image Analysis*, pages 841–851, 2005.

A.K. Jain. *Fundamentals of Digital Image Processing*. Prentice-Hall, Inc., Englewood Cliffs, N.J., USA, 1989.

G.S. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33:82–95, 1971.

R. Lenz. *Group Theoretical Methods in Image Processing*. Springer Verlag, Lecture Notes, 1990.

D.G. Lowe. Distinct image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

C.A. Micchelli and M. Pontil. On learning vector-valued functions. *Neural Computation*, 17:177–204, 2005.

W. Miller. Topics in harmonic analysis with applications to radar and sonar. *IMA Volumes in Mathematics and its Applications*, 1991.

D. Mumford, J. Fogarty, and F. Kirwan. *Geometric Invariant Theory*. Springer, 1994.

L. Nachbin. *The Haar Integral*. D. van Nostrand Company, Inc., Princenton, New Jersey, Toronto, New York, London, 1965.

M. Pontil and C.A. Micchelli. Kernels for multi-task learning. In *Proceeding of the NIPS*, 2004.

V. I. Paulsen R. G. Douglas. *Hilbert Modules over Function Algebras*. Wiley New York, 1989.

M. Reisert and H. Burkhardt. Averaging similarity weighted group representations for pose estimation. In *Proceedings of IVCNZ'05*, 2005.

B. Schoelkopf and A. J. Smola. *Learning with Kernels*. The MIT Press, 2002.

I. Schur. *Vorlesungen ueber Invariantentheorie*. Springer Verlag, Berlin, Heidelberg, New York, 1968.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

Journal of Machine Learning Research 8 (2007) 409-439 Submitted 7/04; Revised 8/06; Published 3/07

Boosted Classification Trees and Class Probability/Quantile Estimation

David Mease MEASE_D@COB.SJSU.EDU
Department of Marketing and Decision Sciences
San Jose State University
San Jose, CA 95192-0069, USA

Abraham J. Wyner AJW@WHARTON.UPENN.EDU
Andreas Buja BUJA@WHARTON.UPENN.EDU
Department of Statistics
Wharton School, University of Pennsylvania
Philadelphia, PA 19104-6340, USA

Editor: Robert Schapire

Abstract

The standard by which binary classifiers are usually judged, misclassification error, assumes equal costs of misclassifying the two classes or, equivalently, classifying at the 1/2 quantile of the conditional class probability function $P[y=1|x]$. Boosted classification trees are known to perform quite well for such problems. In this article we consider the use of standard, off-the-shelf boosting for two more general problems: 1) classification with unequal costs or, equivalently, classification at quantiles other than 1/2, and 2) estimation of the conditional class probability function $P[y=1|x]$. We first examine whether the latter problem, estimation of $P[y=1|x]$, can be solved with LogitBoost, and with AdaBoost when combined with a natural link function. The answer is negative: both approaches are often ineffective because they overfit $P[y=1|x]$ even though they perform well as classifiers. A major negative point of the present article is the disconnect between class probability estimation and classification.

Next we consider the practice of over/under-sampling of the two classes. We present an algorithm that uses AdaBoost in conjunction with **O**ver/**U**nder-**S**ampling and **J**ittering of the data ("JOUS-Boost"). This algorithm is simple, yet successful, and it preserves the advantage of relative protection against overfitting, but for arbitrary misclassification costs and, equivalently, arbitrary quantile boundaries. We then use collections of classifiers obtained from a grid of quantiles to form estimators of class probabilities. The estimates of the class probabilities compare favorably to those obtained by a variety of methods across both simulated and real data sets.

Keywords: boosting algorithms, LogitBoost, AdaBoost, class probability estimation, over-sampling, under-sampling, stratification, data jittering

1. Introduction

Misclassification error is the standard by which binary classifiers are usually judged. This implicitly assumes equal cost of misclassifying the two classes or, equivalently, classifying at the 1/2 quantile of the *conditional class probability function*, or **CCPF** for short. By this standard, AdaBoost and other boosting algorithms have demonstrated some of the lowest misclassification errors across a wide range of applications. Furthermore, boosting algorithms are successful at minimizing misclassification error in ways that are remarkably insensitive to the algorithm's number of iterations. However, for many problems, this is insufficient. Since misclassification error implicitly assumes equal costs for each of the two classes, it is not always an appropriate standard. One is often interested in classification with unequal costs, as in medical problems where a false negative is often much more serious than a false positive. When costs are unequal a desirable classifier selects, for a given feature vector x, the class label that minimizes the expected cost. Alternatively, classification problems may be formulated not in terms of the differential costs of misclassification, but rather in terms of a classifier's ability to predict a quantile threshold, as in marketing when households with an estimated take-rate higher than, say, 0.80 are slated for a campaign. In short, the problem of binary classification with unequal costs is equivalent to the problem of thresholding conditional class probabilities at arbitrary quantiles. As a special case, one obtains the equivalence of an equal-cost classifier and a median classifier that thresholds the conditional probability of both classes at 1/2.

A related problem is that of mapping a classifier to a base rate for the two classes other than that for which the classifier was trained. For example, the training sample may have contained 10% positives, but one is interested in a classifier that performs well when there are 50% positives. A simple calculation shows that a change of base rate is equivalent to a change in the cost ratio and also to a change of the quantile on the class probabilities. These connections are lucidly explained by Elkan (2001), and we will recount them in Section 3.1 in a form suitable for our purposes.

The problem of classifying with arbitrary cost ratios and/or imbalanced base rates has been addressed by a sizable literature. The problem of imbalance has been of particular interest in areas such as text classification (Liu et al., 2002) and bankruptcy prediction (Foster and Stine, 2004) where there often exist very few positives but a vast supply of negatives. One will then try to correct for this imbalance. With respect to boosting, the literature dealing with unequal costs of misclassification is small relative to the huge literature on boosting with respect to (equal) misclassification error. Modifications to boosting and similar algorithms that have been suggested to handle this more general problem include Slipper (Cohen and Singer, 1999), AdaCost (Fan et al., 1999), CSB1 and CSB2 (Ting, 2000) and RareBoost (Joshi et al., 2001). These algorithms have demonstrated some success over straightforward AdaBoost, but no single such algorithm stands out as the favorite.

1.1 Over/Under-Sampling

One general class of approaches (popular in the Computer Science community) for calibrating any equal-loss classifier to handle the equivalent problems of unequal costs, arbitrary quantile thresholds, and imbalanced base rates is the idea of over- and under-sampling (Chan and Stolfo, 1998; Elkan, 2001; Estabrooks et al., 2004). In these schemes one typically over-samples the minority class by sampling with replacement, and/or one under-samples the majority class by sampling without replacement. Sampling with replacement is necessary when the class size is increased, whereas sampling without replacement seems more natural when the class size is decreased. Note that sampling with replacement is bound to produce ties in the sample, especially as the sampling rate increases. An interesting variation of the idea is the Synthetic Minority Over-Sampling TEchnique ("SMOTE") by Chawla et al. (2002, 2003) which avoids ties in the over-sampled minority class by moving the sampled predictor points toward neighbors in the minority class.

The necessity to break ties is an issue that is addressed in detail in Section 3 and also summarized here. Ties can be a problem because classification methods may be driven more by the set of distinct data points than the multiplicities of the tied data points. Changing multiplicities, which is what sampling with replacement does, will therefore have a smaller than expected effect. Thus changing multiplicities should be accompanied by breaking the ties, as in SMOTE. There is evidence that tie-breaking does not need to be sophisticated; random perturbations or "jittering" of the data points in predictor space work surprisingly well. The effectiveness of this technique for AdaBoost is demonstrated in this article.

1.2 Direct Estimation of the CCPF

A second general solution to the problem of classification at arbitrary quantile thresholds is by direct estimation of conditional class probability functions. This is the traditional approach in statistics, as exemplified by logistic regression. Given an estimate of the CCPF one can trivially achieve classification at any arbitrary quantile by thresholding the estimated CCPF at that quantile.

If estimation of a CCPF solves the problem of classification for arbitrary costs, quantiles and imbalances, it is natural to wonder why one bothers to solve the problem by any other means. Why would there still be an interest in, for example, classifiers trained on over/under-sampled training data? The reason for this has to do with the fact that finding a good classifier is a robust endeavor, whereas estimation of a CCPF is a subtle business. The latter has to simultaneously contain the information for good classification at *all* quantiles. By comparison, a classifier is usually described by thresholding some score function at zero, but this function is quite arbitrary except for the requirement to be on the proper side of zero often enough to produce few errors on test samples. This robustness of the classification problem (observed in the literature) is often, but not always, due to the crude nature of the criterion, namely, misclassification error.

Friedman et al. (2000) show that a stagewise approach to the minimization of an exponential loss function by a linear combination of classification trees is algorithmically similar to AdaBoost. In fact, these same authors (in Hastie et al., 2001, Section 10.2) attribute boosting's success to this similarity. This stagewise logistic regression view immediately led to two major developments. First, by connecting AdaBoost to logistic regression, Friedman et al. (2000) and Hastie et al. (2001) implied that one can apply a link function to the scores produced by AdaBoost to provide estimates of the CCPF. Second, by minimizing different loss functions, these authors led the way to the development of new boosting algorithms. In particular, in Friedman et al. (2000) we first encounter *LogitBoost*, an algorithm which computes the CCPF directly by a stagewise minimization of *log* loss on the training data. For the purposes of this paper we will focus on Logitboost; however, it should be noted that other researchers have also developed modifications to AdaBoost for minimization of log loss. One such algorithm is given by Duffy and Helmbold (1999). A similar algorithm is given by Collins et al. (2000) which is motivated by viewing boosting as an optimization of Bregman divergences.

We will give evidence that neither LogitBoost nor the application of the prescribed link to AdaBoost are effective at estimating the CCPF. We will show that when AdaBoost has been run for the large number of iterations required to produce a successful classifier (with respect to misclassification error), its scores have typically diverged to such a degree that putting them through a link function produces values for the CCPF that cluster near 0 and 1. Hence, they are quite erratic as CCPF estimates. The prevailing solution to this problem is regularization of the CCPF estimator. Such regularization methods include the imposition of limits on the class of base learners (i.e., decision stumps instead of trees), the refinement of the learning rate, or simply early stopping. This is not the approach we choose here, though. Instead, our aim is to use off-the-shelf boosting and explore how its manifest success in classification can be carried over to CCPF estimation.

In order to illustrate boosting's inability to estimate CCPF's, we present simulations for which AdaBoost is able to successfully lower the misclassification error for classification trees, but is unable to find the CCPF since its estimates get worse with each iteration of the algorithm. We conclude that AdaBoost is successful at estimating classification *boundaries* even though its scores do not estimate conditional class probabilities. Further, we show that *the same is true of LogitBoost*, despite the fact that estimation of conditional class probabilities is the motivation behind this algorithm. In short, both of these approaches are often ineffective for CCPF estimation.

Because of the relative difficulty of CCPF estimation compared to classification, it will always remain of interest to approach the latter problem directly. In fact, we take this argument as a license for travelling the opposite direction: we construct estimators of CCPF's from collections of classifiers computed from grids of quantiles. The precision of such estimators depends on the denseness of the quantile grids, but in view of the natural sampling variability of CCPF estimators, the granularity of the grid may need only compare with the noise level. Furthermore, if the primary use of the CCPF estimator is thresholding for classification, then the proposed estimator will be satisfactory for practical purposes if the grid contains all quantiles of actual interest. We show experimentally that over- and under-sampled estimates of the CCPF using AdaBoost, constructed in this way, compare favorably with competing methods.

The evidence presented in this paper may seem to be at odds with the logistic regression view of boosting. Certainly, from a practical point of view, this is true to some extent. Our experiments suggest that boosted classification trees produce excellent estimates of class membership and are surprisingly resistant to overfitting, while the same is not true of the values that the logistic regression view of Friedman et al. (2000) would suggest are the estimates of the class probabilities. However, a couple of other factors should be considered here. First, the Friedman et al. (2000) paper is not the only research connecting logistic regression and boosting. For instance, Lebanon and Lafferty (2001) present a very different view of this relationship. Our focus is only on probability estimation using the Friedman et al. (2000) link functions. Secondly, good estimation of the CCPF is not always necessary even for logistic regression itself. Logistic regression is just one of many examples where the underlying model may not represent the true distribution of data well, but can still be quite useful for prediction. Freund and Schapire (2004) discuss this point. They draw a parallel to HMM's for speech analysis and write, "nobody using HMM's for speech analysis really thinks that speech can be synthesized by these HMM's."

In the next section we present the argument that boosted classification trees combined with the Friedman et al. (2000) link functions do not estimate conditional class probabilities. The boosting algorithms we consider are standard, off-the-shelf techniques. It may be argued that some variations of boosting employing extensive regularization (for example, Zhang and Yu 2005, Rosset et al. 2004 and Blanchard et al. 2003) or direct cost-optimization techniques can overcome this problem. We do not consider these techniques in this paper for a number of reasons. A practical reason is that these techniques are often considered not worth the trouble by many practitioners. This is especially true when a simple effective alternative exists, as we will demonstrate. Further, the focus of the paper is on how the off-the-shelf utility of boosting algorithms which greatly contributes to their importance can be carried over to CCPF estimation in a straightforward manner. Regularization and cost-optimization boosting algorithms are outside the scope of this paper.

2. Boosted Classification Trees Do Not Estimate Conditional Class Probabilities

We now demonstrate, by means of simulations, how boosting with classification trees fails to estimate the CCPF while at the same time performing well in terms of the classification criterion, namely, misclassification error.

We introduce customary notation. We are given training data $x_1, ..., x_n$ and $y_1, ..., y_n$ where each x_i is a $d-$dimensional vector of predictors and $y_i \in \{-1, +1\}$ is the associated observed class label. To justify generalization, it is usually assumed that training data as well as any test data are *iid* samples from some population of (x, y) pairs.

We will consider the two boosting algorithms *LogitBoost* (Friedman et al., 2000) and (discrete) *AdaBoost*. AdaBoost (Freund and Schapire, 1996) is the most common implementation of boosting. The algorithm is as follows. First let $F_0(x_i) = 0$ for all x_i and initialize weights $w_i = 1/n$ for $i = 1, ..., n$. Then repeat the following for m from 1 to M:

- Fit the classifier g_m to the training data using weights w_i where g_m maps each x_i to -1 or 1.
- Compute the weighted error rate $\varepsilon_m \equiv \sum_{i=1}^n w_i \mathrm{I}\{y_i \neq g_m(x_i)\}$ and half its log-odds, $\alpha_m \equiv \frac{1}{2} \log \frac{1-\varepsilon_m}{\varepsilon_m}$.
- Let $F_m = F_{m-1} + \alpha_m g_m$.
- Replace the weights w_i with $w_i \equiv w_i e^{-\alpha_m g_m(x_i) y_i}$ and then renormalize by replacing each w_i by $w_i / (\sum w_i)$.

The final classifier is 1 if $F_M > 0$ and -1 otherwise. It has been observed repeatedly that the performance of the procedure, with respect to misclassification error, is quite insensitive to the choice of M and tends to result in small error rates (relative to competing methods) across a wide range of applications, especially in high dimensions. In many real examples, large values of M work very well.

Motivated by analogies between AdaBoost and additive logistic regression, Friedman et al. (2000) proposed a connection between the score function $F_m(x)$ and a logistic regression model. They suggest that an estimate $p_m(x)$ of the CCPF $p(x)$ can be obtained from F_m through an (essentially) logistic link function:

$$p_m(x) = p_m(y = 1|x) = \frac{1}{1 + e^{-2F_m(x)}}. \qquad (1)$$

In fact, using this link function it can be shown that the "exponential loss" of AdaBoost can be mapped to a scoring rule on the probabilities similar to a maximum likelihood criterion (Buja et al., 2006). From this perspective, each iteration of boosting uses the current estimates of the probabilities to minimize this criterion in a stagewise manner.

The view of boosting as a form of logistic regression and thus as an estimator of the CCPF has given rise to much research since its publication. It has been the basis for recent work on choosing stopping rules so as to not overfit probabilities (Dettling and Buhlmann, 2003) as well as efforts to establish consistency using regularization (Bickel et al., 2006; Jiang, 2004; Lugosi and Vayatis, 2004). Furthermore, in their seminal article, Friedman et al. (2000) introduced LogitBoost, which directly minimizes the negative Bernoulli log-likelihood in a stagewise manner, instead of the exponential loss of AdaBoost. This is the prevailing reason why LogitBoost is widely thought to be a better estimator of the CCPF. LogitBoost is the second algorithm we will examine in our simulations.

The statistical view that casts the outputs from boosting classification trees as estimates of the CCPF poses no problem for the purpose of median classification. The symmetry of the chosen link functions implies that thresholding p_m at 1/2 amounts to thresholding F_m at 0. Used in this way, boosting provides excellent classification rules that are quite insensitive to the number of iterations M. However, as we will see in the following simulations, the idea that boosting classification trees can be used to estimate conditional probabilities is questionable. While in most instances there is little to no overfitting with respect to misclassification error (i.e., median estimation), with respect to probability estimation overfitting is likely and often very pronounced. This sheds doubt on the idea that the success of boosting is due to its similarity to logistic regression, and research motivated by this connection, while insightful in its own right, is most likely mistaken if it claims to throw light on boosting.

In all our experiments the base learners will be 8-node trees, as in Friedman et al. (2000), unless otherwise noted. The two exceptions are one simulated data set for which we will consider stumps in addition to 8-node trees for sake of comparison, and one large data set for which we use 2^{10}-node trees. The AdaBoost algorithm we use is described above, and the classification trees are constructed using the function "Rpart" in the R software package (http://www.r-project.org/). The probabilities are obtained by mapping the resulting score $F_m(x)$ through the link (1) to produce the CCPF estimate $p_m(y = 1|x)$. We will refer to this procedure as *P-AdaBoost*. We will use Dettling and Buhlmann's (2003) implementation of LogitBoost with two minor modifications to the source code to allow the function Rpart to fit 8-node decision trees instead of stumps.

2.1 A 2-dimensional Circle Model

For our first simulation, we consider the domain of X to be the square $[0,50]^2$ and construct level curves of $p(x)$ to be concentric circles with center $(25,25)$. We let

$$p(x) = p(y = 1|x) = \begin{cases} 1 & r(x) < 8 \\ \frac{28-r(x)}{20} & 8 \leq r(x) \leq 28 \\ 0 & r(x) > 28 \end{cases}$$

where $r(x)$ is the distance from x to the point $(25,25)$ in R^2. We will call this the *2-Dimensional Circle* model.

The right panel of Figure 1 shows these probabilities for a hold-out sample that is a grid of 2500 evenly spaced points on $[0,50]^2$. The points are color-coded such that the white region corresponds to the subregion of the square where $p(x) \leq 0.1$. The pink subregion is where $0.1 < p(x) \leq 0.5$, the red subregion is where $0.5 < p(x) \leq 0.9$ and the black subregion is where $p(x) > 0.9$. For the training data, we simulated $n = 1000$ *iid* observations with x uniform on $[0,50]^2$ and the class label y chosen according to $p(y = 1|x)$ above. The training data is shown in the left panel of Figure 1 colored such that points are green plus signs when $y = -1$ and black plus signs when $y = 1$.

If the algorithms are stopped early enough, both P-AdaBoost and LogitBoost provide reasonably accurate estimates of the CCPF. For example, we present in Figure 2 the resulting estimates of $p_m(x)$ at $m = 5$ iterations. We use the same color coding as Figure 1. It is clear that both P-AdaBoost (left panel of Figure 2) and LogitBoost (right panel of Figure 2) estimate conditional probabilities that match the true probabilities (the right panel of Figure 1) fairly well. However, as the number of iterations increases, both begin to overfit the probabilities. As m increases the resulting conditional probabilities converge to zero and one for all x, and consequently the estimates of all quantiles

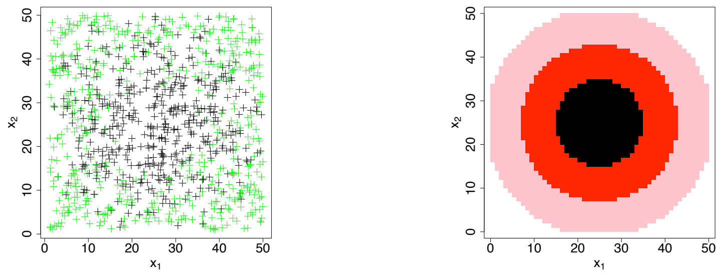


Figure 1: Training data (left panel) and true probabilities (right panel) for the 2-Dimensional Circle model with $n = 1000$.

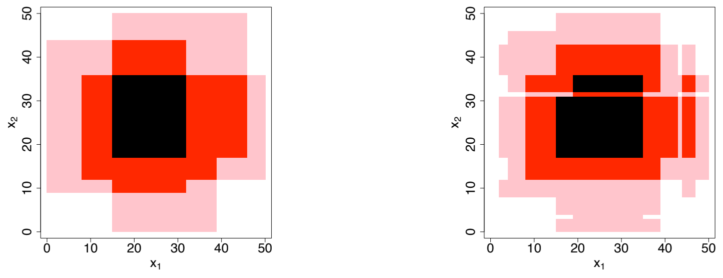


Figure 2: Estimated probabilities for P-AdaBoost (left panel) and LogitBoost (right panel) at 5 iterations.

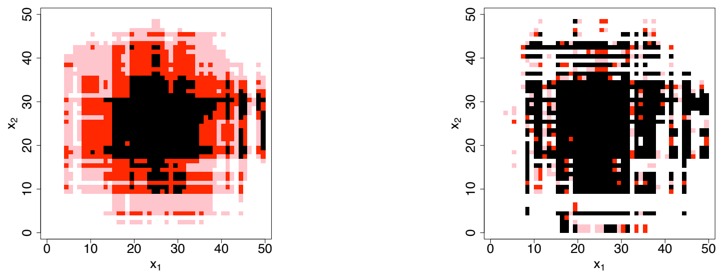


Figure 3: Estimated probabilities for P-AdaBoost (left panel) and LogitBoost (right panel) at 800 iterations.

converge to the estimate for the median. Figure 3 reveals the conditional probability estimates for the hold-out sample at $m = 800$ iterations. At $m = 800$, the P-AdaBoost algorithm exhibits some overfitting, as the white and black regions are larger than they should be. LogitBoost in contrast, has overfit the data more severely at $m = 800$. For almost all x in the hold-out sample, its estimate of the CCPF is either greater than 0.9 (black) or less than 0.1 (white). In fact, as m increases further both P-AdaBoost and LogitBoost eventually produce estimates of the CCPF in which almost all probability estimates are near zero or one. Thus, these algorithms eventually provide a complete overfit of the probabilities. In contrast, the estimate of the median remains relative stable as m increases and is quite accurate.

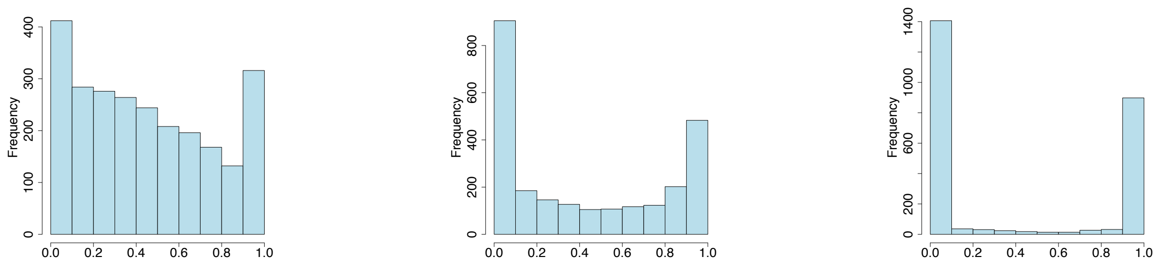


Figure 4: Histograms of true probabilities (first plot) and estimated probabilities for P-AdaBoost (second plot) and LogitBoost (third plot) at 800 iterations.

Overfitting the data, which is reflected in the divergence of the probability estimates to zero and one by both P-AdaBoost and LogitBoost, is particularly pronounced in this example since many of the true probabilities are actually quite far from zero and one. This is shown in Figure 4. The left most panel is a histogram of the true CCPF over the 2500 randomly chosen points x that form the hold-out sample. The middle and right most panels are the histograms of the estimated CCPF for P-AdaBoost and LogitBoost respectively, again at 800 iterations, on the hold-out sample. It is clear, in the case of LogitBoost, that for almost every x the estimated CCPF is greater than 0.9 or less than 0.1. In contrast, the true CCPF is much more spread out. The histogram of the estimated probabilities for P-AdaBoost exhibits the same extremal clumping, although not as severely as LogitBoost.

2.2 A 10-dimensional Model

Next we borrow a model first introduced in Friedman et al. (2000) that reveals the same pattern of overfitting and the same divergence of the CCPF. This example demonstrates how overfitting is possible in cases where the Bayes error rate is near or even equal to zero.

For the simulation, we sample n observations of the 10-dimensional vector x generated *iid* from $N^{10}(0,I)$ (we call this the *10-Norm* model). The CCPF is again the conditional probability that $y=1$ given x (which we denote $p(x)$) and is defined as follows:

$$\log\left(\frac{p(x)}{1-p(x)}\right) = \gamma[1-x^{(1)}+x^{(2)}-x^{(3)}+x^{(4)}-x^{(5)}+x^{(6)}]\sum_{j=1}^{6} x^{(j)} \tag{2}$$

where the superscripts denote the components of the vector. As γ increases the Bayes error rate decreases. We begin with $\gamma = 10$, which mimics the simulation in Friedman et al. (2000) exactly. With $\gamma = 10$ the Bayes error rate is only 3%. When $\gamma = 0.5$ the noise level in the model is higher and the Bayes error rate climbs to 22%.

Since these simulations are in ten dimensions instead of two, it is difficult to display the overfitting of the probabilities graphically as in the previous example. Instead we consider some quantitative measures for the accuracy of probability estimates. There is no single, established method for measuring the performance of class probability estimators (see Zadrozny and Elkan (2001) for a discussion). When simulating from models with known parameters, there are many choices. We choose the following loss functions which all provide a penalty depending on the extent to which the estimate of the probability $\hat{p}$ diverges from the true probability $p = p(x)$ on a hold-out sample $x_1^*, \ldots, x_{n^*}^*$.

- Squared Loss:

$$\sum_{i=1}^{n^*} [p(x_i^*) - \hat{p}(x_i^*)]^2 \tag{3}$$

- Log Loss:

$$-\sum_{i=1}^{n^*} [p(x_i^*) \log(\hat{p}(x_i^*)) - (1 - p(x_i^*)) \log(1 - \hat{p}(x_i^*))] \tag{4}$$

- Exponential Loss:

$$\sum_{i=1}^{n^*} \left[p(x_i^*) \sqrt{\frac{1 - \hat{p}(x_i^*)}{\hat{p}(x_i^*)}} + (1 - p(x_i^*)) \sqrt{\frac{\hat{p}(x_i^*)}{1 - \hat{p}(x_i^*)}} \right] \tag{5}$$

This last scoring function is the population analogue of the exponential loss of AdaBoost when composed with the link (1) suggested by Friedman et al. (2000) as pointed out by Buja et al. (2006). The log loss is the population version of the negative log likelihood of the Bernoulli model. For the simulations we choose a hold-out sample with $n^* = 2500$ drawn from the same distribution as the training data. Note, however, that since both log loss and exponential loss are unbounded, we threshold $\hat{p}$ so that it always lies between 0.05 and 0.95 before computing these two loss functions. This prevents a few conditional probability estimates that are close to zero or one from dominating the loss. This truncation will become more important in Section 3.

The four plots in Figure 5 display the values of misclassification error, squared loss, log loss and exponential loss averaged over the hold-out sample for the 10 dimensional normal (10-Norm) model given by Equation (2) with $n = 500$ (and $\gamma = 10$). The large black dot at zero iterations is the loss occurred by estimating all probabilities by the marginal sample proportion in the training data or analogously by the majority class in the training data for misclassification error. P-AdaBoost is represented by the black line and LogitBoost by the blue line. The red and green lines represent the performance of JOUS-Boost, that is, AdaBoost combined with over/under-sampling and jittering, as will be described in Sections 3 and 4. The graph for misclassification error depicts the trademark features of boosting algorithms. There seems to be little or no overfitting and the error curve reflects a general decreasing trend as more and more iterations are performed. The classifiers continue to improve their performance on hold-out samples even as the number of iterations increase beyond the number required to fit the training data perfectly (that is, to achieve zero misclassification error on the training data). Friedman et al. (2000) also observed this trend; however, they did not examine any measures of performance with respect to the probability estimation. From the other three plots that graph the performance of the CCPF estimator, we can see that all three of the measures display a generally increasing pattern, implying that both of the boosting algorithms overfit the probabilities, even though they do not overfit with respect to out-of-sample misclassification error. Figure 6 shows that this pattern becomes even more pronounced when the value of γ is decreased from 10 to 0.5, which increases the Bayes error rate from 0.03 to 0.22.

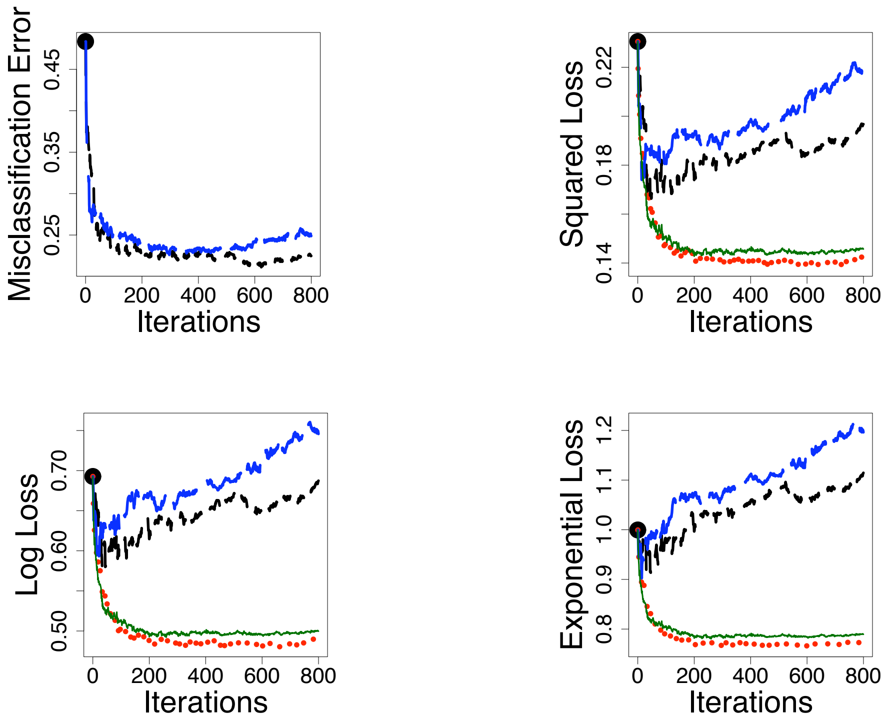


Figure 5: Misclassification error, squared loss, log loss and exponential loss for the 10-Norm model with $n = 500$ and $\gamma = 10$. **Black (Short Dashes)**: P-AdaBoost. **Blue (Long Dashes)**: LogitBoost. **Red (Dotted)**: JOUS-Boost, over-sampled. **Green (Solid)**: JOUS-Boost, under-sampled.

The observed divergence of the probability estimates to extreme values is a consequence of the fact that the span of the space of weak learners (8-node trees in this case) is rich enough to separate the two classes completely. This is always the case when the algorithms use all possible classification trees of a certain depth defined on the training data. As the training set grows, the space of weak learners grows (since they are data defined), allowing every point in the training set to be correctly classified. This effect holds even if the trees are restricted to a depth of 2 (so-called stumps), although the number of iterations required is typically a lot larger. More discussion of separation with boosting and a proof that boosted stumps will eventually separate all the data (for the case of a single predictor) can be found in Jiang (2000). The only condition needed for the separability is that at least one of the predictors assumes unique values for all points. For a continuous predictor this will occur with probably one, but for the case of all categorical predictors separation may not be possible.

As a consequence of this separability, as the number of boosting iterations increases, the scores $F(x_i)$ for which $y_i = +1$ tend towards $+\infty$, and scores for which $y_i = -1$ tend toward $-\infty$. This occurs because the exponential loss (or negative log likelihood in the case of LogitBoost) is minimized at these limiting values. On hold-out samples we observe that when $y = +1$, boosting usually returns a very large value for F, thus resulting in a correct classification when the threshold for F

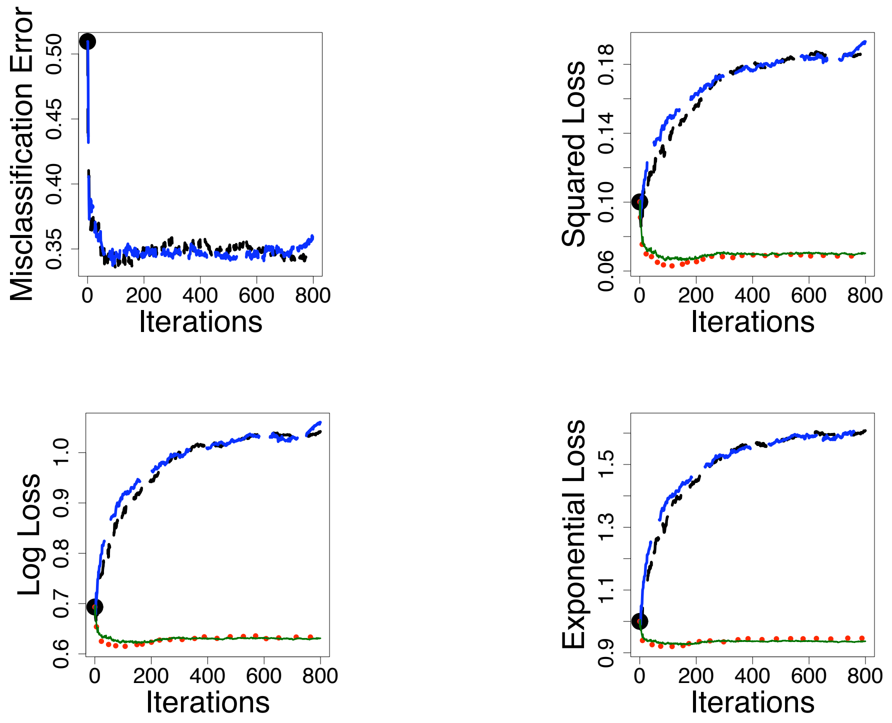


Figure 6: Misclassification error, squared loss, log loss and exponential loss for 10-norm model with $n = 500$ and $\gamma = 0.5$. **Black (Short Dashes)**: P-AdaBoost. **Blue (Long Dashes)**: LogitBoost. **Red (Dotted)**: JOUS-Boost, over-sampled. **Green (Solid)**: JOUS-Boost, under-sampled.

is zero. The analogue is true for observations for which $y = -1$. The effectiveness of boosting for classification problems is not in spite of this phenomenon, but rather due to it. The fact that the absolute values of F_m grow without bound as m increases does not necessarily lead to an over-fit with regard to misclassification error because the absolute value is irrelevant to a median-classifier; only the sign matters.

If boosting is to be used as a probability estimator or a quantile classifier by thresholding at a value other than zero (as prescribed by a link such as that of Equation (1)), then the absolute value of F_m *does* matter. Boosting will eventually estimate $p(x)$ by either 0 or 1. The specific form of the monotonic link function is irrelevant, so long as it ascends from zero to one, because the absolute value of F_m is largely a function of m rather than a reflection on the true class probability. We illustrate this graphically by tracing the values of F_m for AdaBoost in our last simulation for the points in the hold-out sample for which the true CCPF is between 0.7 and 0.8. Using the link function in Equation (1) we can compute that the fitted values of F_m for these points should ideally lie between 0.42 and 0.69. However, in Figure 7 we see that the median of F_m for the hold-out points with true p between 0.7 and 0.8 has a general linear trend, and by $m = 800$ iterations is greater than 6, corresponding to a $p(x)$ greater than 0.99999. Without the thresholding at 0.95 and 0.05 the overfitting caused by such extreme probabilities would be even more pronounced than displayed in the plots. This unbounded linear growth of the score function F_m suggests that we can obtain any estimate greater than 1/2 for the CCPF at these points by simply choosing the appropriate stopping time M. The link function is irrelevant.

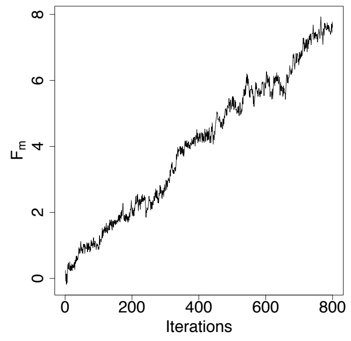


Figure 7: Median$(F_m(x) : 0.7 \leq p(x) \leq 0.8)$

As mentioned earlier, the realization that boosting often overfits the CCPF but not the median value of the CCPF has many important implications. It brings into question whether it is right to attribute the success of boosting to its similarity to logistic regression. Furthermore, the practical work that has arisen from this theoretical view of boosting may be misguided. For instance, stopping rules that are based on cross-validation using a maximum likelihood criterion are questionable. In fact, Dettling and Buhlmann's version of LogitBoost, which was modified for use in the previous simulations, has the option of employing such a stopping rule. Interestingly the authors themselves remark that for gene expression applications, the maximum likelihood criterion indicated that stopping should be implemented for some $M < 100$, but they observe that early stopping does not seem to provide any significant advantages for classification and "most often yields slightly worse results" (Dettling and Buhlmann, 2003).

Before leaving this section we should point out that there do exist cases in which overfitting of the CCPF does not appear to occur (at least not for $m \leq 800$). As discussed above, the probability estimates from boosting will necessarily eventually diverge to zero and one. This is true as long as the space of weak learners is allowed to increase in complexity with the sample size. However, the extent to which this divergence causes overfitting depends on the true probabilities and the size of the training data. In fact, the original simulation by Friedman et al. (2000) used a large data set ($n = 5000$) and $\gamma = 10$, implying a Bayes error rate of 0.03. With these values for the parameters, there does not seem to be substantial overfitting of the probabilities (by $m = 800$). To see this, consider the plots in Figure 8. The success in the problem can be attributed to the large sample size, the moderate number of dimensions (10) and the low Bayes error rate (0.03). Because the true probabilities are practically zero or one, the large number of training points in few dimensions allows for very accurate estimation of the true CCPF values that happen to clump near zero and one.

2.3 Simulations under Greater Uncertainty

It is quite possible that overfitting of the CCPF occurs even with a large sample of training data, namely, when either 1) the Bayes error is high, or 2) the number of dimensions is large, or 3) the decision boundaries are very complex. In the 10-Norm model we can create an example with higher

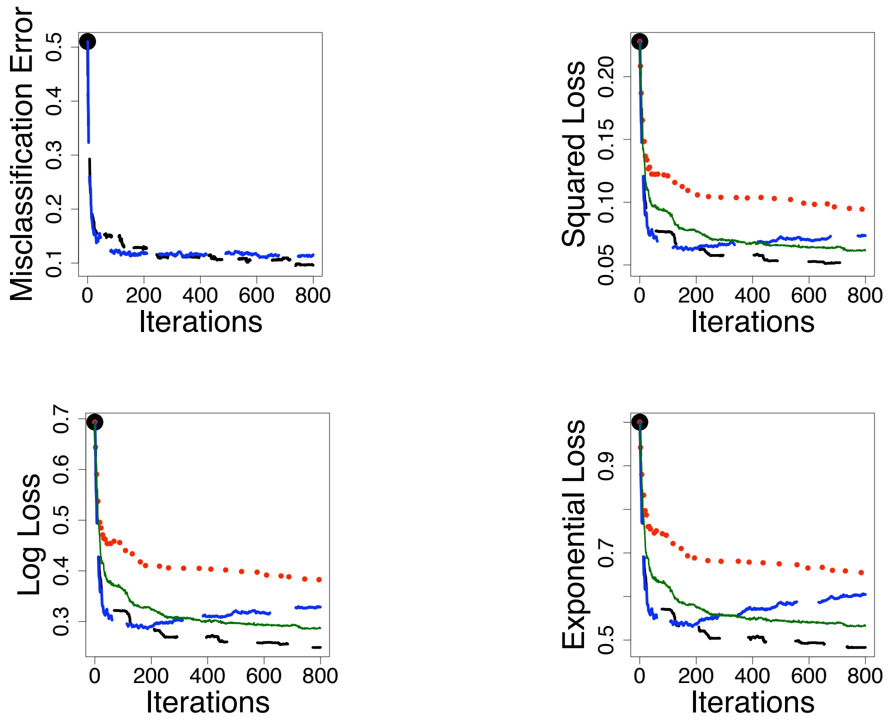


Figure 8: 10-Norm model with $\gamma = 10$ (Bayes error 0.03) and $n = 5000$. **Black (Short Dashes)**: P-AdaBoost. **Blue (Long Dashes)**: LogitBoost. **Red (Dotted)**: JOUS-Boost, over-sampled. **Green (Solid)**: JOUS-Boost, under-sampled.

Bayes error by lowering γ. Figure 9 displays the results for the 10-Norm simulation with $n = 5000$ and $\gamma = 0.5$, corresponding to a Bayes error of 0.22. Here we see that LogitBoost clearly overfits the probabilities, although P-AdaBoost does not, at least not by $m = 800$ iterations.

Figure 10 shows a simulation from the model

$$p(x) = 0.03 + 0.94\, \mathrm{I}\left\{ \sum_{j=1}^{d} x^{(j)} > 0 \right\} \qquad (6)$$

in $d = 1000$ dimensions. We refer to this as the *1000-dim* model. The Bayes error rate is 3%. We chose a sample of size $n = 5000$. Note that both the sample size and the Bayes error rate match those used in original simulation by Friedman et al. (2000) with the 10-Norm model. Our experiments show that both P-AdaBoost and LogitBoost overfit the probabilities from the outset—there is no early stopping point which improves the fit. In contrast, the rate of misclassification drops continuously as the number of iterations increases, and there is no evidence of overfitting. Once again, the behavior we observe conflicts with the logistic regression view of boosting.

In cases such as these, a natural question to ask is whether the overfitting with respect to the probabilities might be avoided by choosing a weaker set of base learners, such as stumps instead of the 8-node trees. In fact, the logistic regression view of boosting suggests stumps are particularly attractive for a model in which the Bayes decision boundary is additive, as is the case for the

1000-dim model considered in the previous paragraph. The reasoning is that since stumps involve only single predictors, the boosted stumps in turn yield an additive model in the predictor space. Discussion of this can be found in Hastie et al. (2001) on pages 323-324 and in Friedman et al. (2000) on pages 360-361. Despite this, it can be observed in Figure 11 that using stumps does not eliminate the overfitting problem for the 1000-dim simulation model in the previous paragraph. Just as we observed in Figure 10 with the 8-node trees, the squared loss, log loss and exponential loss for the stumps all begin increasing with the very first iteration and at no point during the 800 iterations even match the majority vote classifier represented by the solid black dot at iteration zero. The use of stumps instead of 8-node trees in this case slows the overfitting, but by no means produces a useful algorithm for estimation of the CCPF. Further, the first plot in Figure 11 also reveals that the resulting classification rule after 800 iterations is not quite as accurate as with the 8-node trees used in Figure 10.

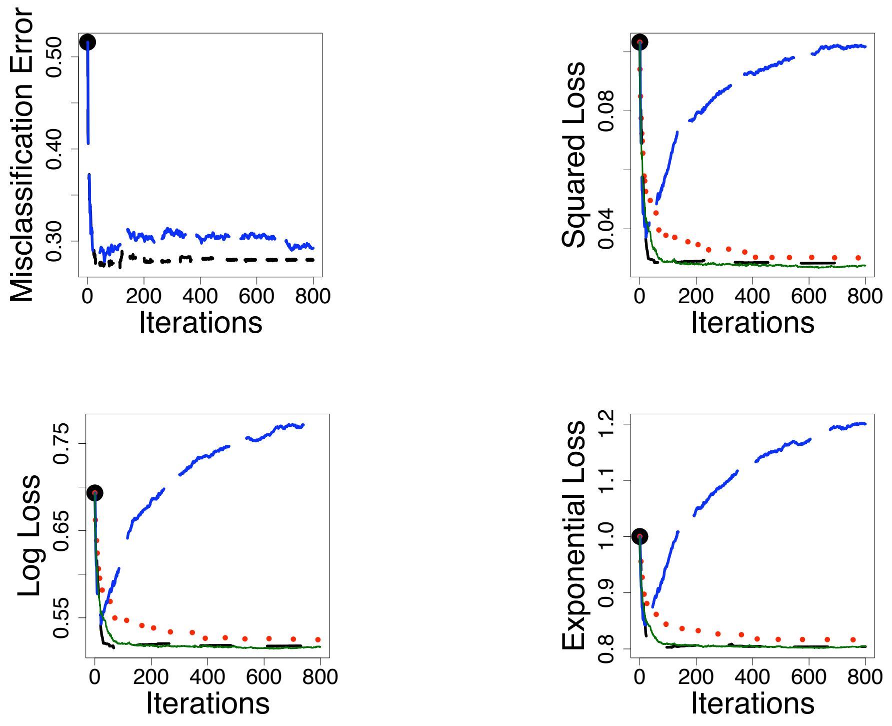


Figure 9: 10-Norm model with $\gamma = 0.5$ (Bayes error 0.22) and $n = 5000$. **Black (Short Dashes)**: P-AdaBoost. **Blue (Long Dashes)**: LogitBoost. **Red (Dotted)**: JOUS-Boost, over-sampled. **Green (Solid)**: JOUS-Boost, under-sampled.

Finally, from the realization that boosting does not provide reliable estimates of quantiles other than the median, we conclude that there is a problem when attempting to use boosting for classification if the misclassification costs of the two classes are unequal, or if the base rate of the training data differs from that of the population of interest as discussed previously. In what follows we address the question of how to use boosting to estimate quantiles other than the median in order to provide solutions to these problems.

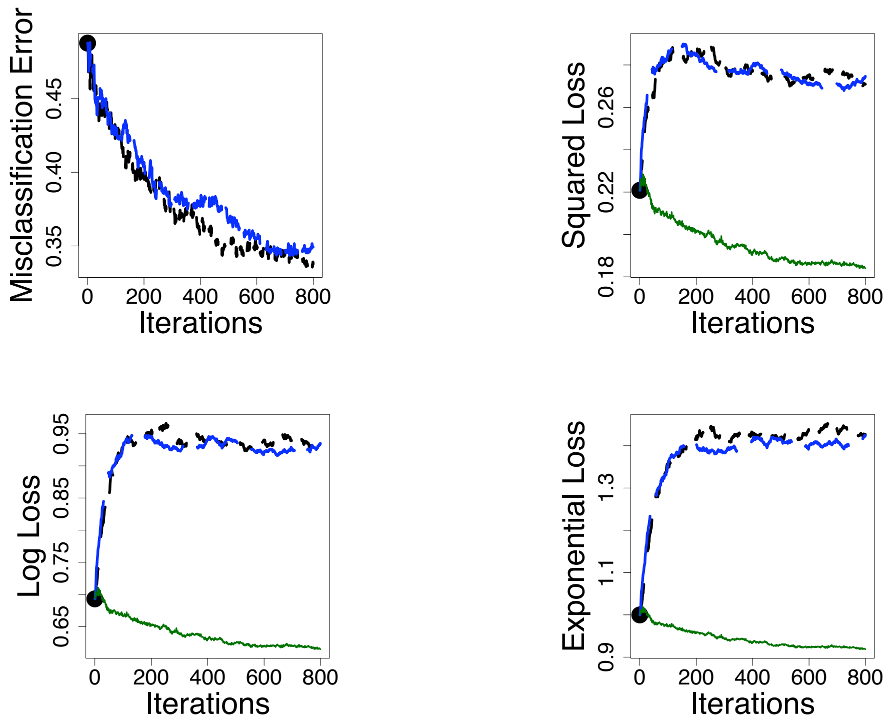


Figure 10: 1000-dim model (Bayes error 0.03) with $n = 5000$. **Black (Short Dashes)**: P-AdaBoost. **Blue (Long Dashes)**: LogitBoost. **Green (Solid)**: JOUS-Boost, under-sampled.

3. Classification With Unequal Costs

Symmetric 0-1 loss is historically motivated by a supposition (previously common in AI, not in statistics) that there always exists a “true” class label, that is, the class probability is either 0 or 1. Hence the goal is always perfect classification, and 0-1 loss is a reasonable measure. In practice, however, it is common to find situations where unequal loss is appropriate and cost-weighted Bayes risk should be the target.

It is an elementary fact that minimizing loss with unequal costs is equivalent to estimating the boundaries of conditional class-1 probabilities at thresholds other than 1/2. We outline the textbook derivation, drawing intuition from the Pima Indians Diabetes Data of the UCI ML Database: Let $p(x) = P[Y = +1|x]$ and $1 - p(x) = P[Y = -1|x]$ be the conditional probabilities of diabetes $(+1)$ and non-diabetes (-1), respectively, and assume the cost of misclassifying a diabetic is $1 - c = 0.90$ and the cost of misclassifying a non-diabetic is $c = 0.10$. (The benefit of scaling the costs to sum to 1 will be obvious shortly.) The following consideration is conditional on an arbitrary but fixed predictor vector x, hence we abbreviate $p = p(x)$. The risk (=expected loss) of classifying as a diabetic is $(1 - p)c$ and the risk of classifying as a non-diabetic is $p(1 - c)$. The cost-weighted Bayes rule is then obtained by minimizing risk: Classify as diabetic when $(1 - p)c < p(1 - c)$ or, equivalently, $c < p$, here: $0.1 < p$, which is what we wanted to show. It is a handy convention to

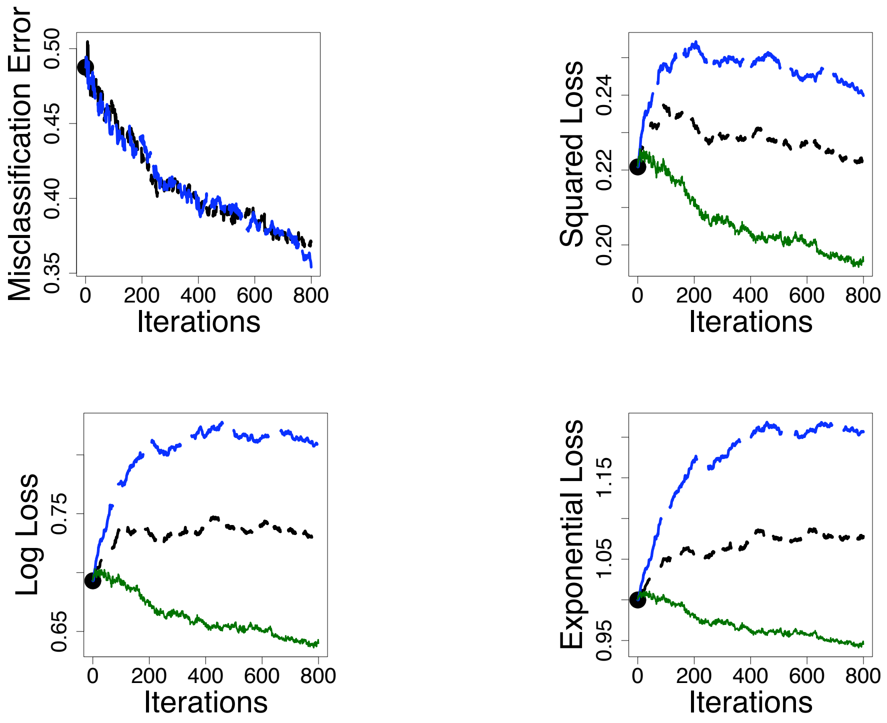


Figure 11: 1000-dim model (Bayes error 0.03) with $n = 5000$ using stumps instead of 8-node trees. **Black (Short Dashes)**: P-AdaBoost. **Blue (Long Dashes)**: LogitBoost. **Green (Solid)**: JOUS-Boost, under-sampled.

scale costs so they add up to 1, and hence the cost c of misclassifying a negative (-1) equals the optimal threshold q on the probability $p(x)$ of a positive ($+1$).

The previous discussion implies the equivalence between the problem of *classification with unequal costs* and what we call the problem of *quantile classification*, that is, estimation of the region $p(x) > q$ in which observations are classified as $+1$. Most classifiers by default (AdaBoost included) assume equal costs, $c = 1-c = 0.5$, which implies that they are *median-classifiers*: $p(x) > q = 1/2$.

3.1 Over/Under-Sampling

Since boosting algorithms are very good median classifiers, one solution is to leave the boosting algorithm as is and instead tilt the data to force the q-quantile to become the median. Simple over/under-sampling, also called stratification, can convert a median classifier into a q-classifier as follows:

Over/under-sampled Classification:

1. Let N_{+1} and N_{-1} be the marginal counts of labels $+1$ and -1. Choose $k_{+1}, k_{-1} > 0$ for which $\frac{k_{+1}}{k_{-1}} = \frac{N_{+1}}{N_{-1}} / \frac{q}{1-q}$.

2. Select k_{+1} observations from the training set for which $y_i = 1$ such that each observation has the same chance of being selected.
3. Select k_{-1} observations for which $y_i = -1$ such that each observation has the same chance of being selected.
4. Obtain a median classifier from the combined sample of $k_{+1} + k_{-1}$ points. Assume its output is a score function $F_m(x)$ such that $F_m(x) > 0$ estimates $p(x) > 1/2$.
5. Estimate x as having $p(x) > q$ if $F_m(x) > 0$.

Note here that for $k_{\pm 1} < N_{\pm 1}$ (i.e., under-sampling) the selection can be done by random sampling with or without replacement, and for $k_{\pm 1} > N_{\pm 1}$ (i.e., over-sampling) the selection can be done either by sampling with replacement or by simply augmenting the data by adding replicate observations.

For the reader's convenience, we briefly give the justification for this re-weighting/re-sampling scheme. It can be inferred from Elkan (2001). We work conveniently in terms of populations/distributions as opposed to finite samples. Let X be the predictor random vector and Y the binary response random variable with values in $\{+1, -1\}$. Let further

$$\begin{aligned} p(x) &= P[Y = +1|x] \, , \\ \pi &= P[Y = +1] \, , \\ f_{+1}(x) &= P[x|Y = +1] \, , \\ f_{-1}(x) &= P[x|Y = -1] \, , \end{aligned}$$

which are, respectively, the conditional probability of $Y = +1$ given $X = x$, the unconditional probability of $Y = +1$, and the conditional densities of $X = x$ given $Y = +1$ and $Y = -1$. The densities $f_{\pm 1}(x)$ describe the distributions of the two classes in predictor space. Now the joint distribution of (X, Y) is given by

$$p(y = +1, x) = f_{+1}(x)\,\pi \, , \quad p(y = -1, x) = f_{-1}(x)\,(1 - \pi) \, ,$$

and hence the conditional distribution of $Y = 1$ given $X = x$ is

$$p(x) = \frac{f_{+1}(x)\,\pi}{f_{+1}(x)\,\pi + f_{-1}(x)\,(1 - \pi)} \, ,$$

which is just Bayes theorem. Equivalently we have

$$\frac{p(x)}{1 - p(x)} = \frac{f_{+1}(x)\,\pi}{f_{-1}(x)\,(1 - \pi)} \, ,$$

or

$$\frac{p(x)}{1 - p(x)} \Big/ \frac{\pi}{1 - \pi} = \frac{f_{+1}(x)}{f_{-1}(x)} \, . \tag{7}$$

Now compare this situation with another one that differs only in the mix of class labels, call them π^* and $1 - \pi^*$, as opposed to π and $1 - \pi$. Denote by $p^*(x)$ the conditional probability of $Y = 1$ given $X = x$ under this new mix. From Equation (7) it is obvious that $p(x)$ and $p^*(x)$ are functions of each other, best expressed in terms of odds:

$$\frac{p^*(x)}{1 - p^*(x)} = \frac{p(x)}{1 - p(x)} \cdot \frac{1 - \pi}{\pi} \cdot \frac{\pi^*}{1 - \pi^*} \, .$$

Obviously thresholds q on $p(x)$ and q^* on $p^*(x)$ transform the same way. If $\pi/(1-\pi)$ is the original unconditional ratio of class labels, we ask what ratio $\pi^*/(1-\pi^*)$ we need for mapping the desired q to $q^* = 1/2$ or, equivalently, $q/(1-q)$ to $q^*/(1-q^*) = 1$. The answer is given by the condition

$$1 \;=\; \frac{q^*}{1-q^*} \;=\; \frac{q}{1-q}\cdot\frac{1-\pi}{\pi}\cdot\frac{\pi^*}{1-\pi^*}\ .$$

Solving for $\pi^*/(1-\pi^*)$, the desired marginal ratio of class labels is

$$\frac{\pi^*}{1-\pi^*} \;=\; \frac{\pi}{1-\pi}\Big/\frac{q}{1-q}\ ,$$

which justifies the algorithm above.

3.2 Difficulties With Over- and Under-Sampling

In principle, both the over- and under-sampled boosting algorithms should be effective q-classifiers for applications in which standard boosting is an effective classifier. However, there are some difficulties with each.

The problem with under-sampling is straightforward: a portion of data from the minority class is not used and consequently the effective sample size is smaller than the actual number of instances in the training data. For training data with a large number of observations, this may not be problematic. However, for small data sets or cases in which the desired quantile is close to zero or one, this can pose difficulties. In such cases, over-sampling may be preferable, although there have been attempts at more elaborate under-sampling techniques which use all the data (Chan and Stolfo, 1998).

Over-sampling has a less obvious problem that is particular to boosting and tends to show up after a large number of iterations. The algorithm is an effective q-classifier after a small number of iterations; however, just as with estimating probabilities using the link function, as the number of iterations increases it tends to revert to a median classifier ($p(x) > 1/2$). The reason for this limiting behavior has to do with the fact that the augmented training set includes many replicates of the same (x,y) pair. As boosting re-weights the training samples, the tied points all change their weights jointly, and their multiplicity is invisible in the iterations because only the sum of their weights is visible. In effect, boosting with over-sampled training data amounts to initializing the weights to values other than $1/n$, but nothing else. For increasing numbers of iterations, the effect of the initial weights diminishes and disappears in the limit.

For illustration we return to the two-dimensional circle example using a sample size of $n = 1000$. We will attempt to estimate the quantile $q = 0.9$ with over-sampled AdaBoost by augmenting the training data with 8 more replicates of each x_i for which $y_i = -1$ ($k_{+1} = N_{+1}$ and $k_{-1} = 9N_{-1}$). The left panel of Figure 12 shows the estimate of this quantile on the hold-out sample after 50 iterations. It appears to be a reasonable estimate of the true quantile which the reader should recall is given by the boundary of the red and black in the right panel of Figure 1. However, the middle panel of Figure 12 shows the estimate for the same simulation after 1000 iterations. In this plot it is clear that the circle has grown larger and has begun to revert back to a median classifier.

One solution to this problem is to add a small amount of random noise to the predictor x so that the replicates are not duplicated. While this *jittering* does add undesirable noise to the estimates of the level curves of the function $p(x)$, the amount of noise needed to alleviate this problem is not large and will only shift the boundary of the Bayes rule in the predictor space by a small amount.

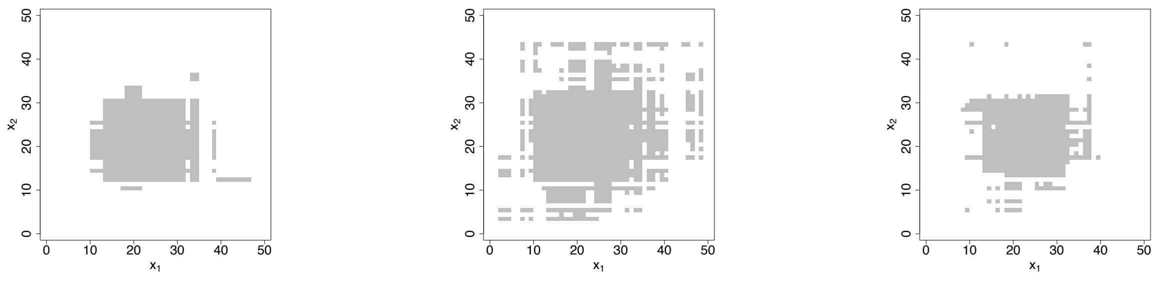


Figure 12: Estimates of the $q = 0.90$ quantile for the circle example. Left panel is $m = 50$, middle panel is $m = 1000$, and right panel is jittering applied for $m = 5000$.

We have achieved good results by adding *iid* uniform $(-\nu\sigma_j, \nu\sigma_j)$ noise to each of the d predictors where σ_j is the standard deviation for the jth predictor in the data set and ν is a tuning parameter that can be chosen to optimize performance. Good values will depend on the size of the training data, sample size, the number of dimensions, noise level, etc. Also, if desired one could consider using more complex procedures to break ties, such as the "SMOTE" algorithm (Chawla et al., 2002, 2003) described before.

As mentioned earlier, we use the acronym "JOUS-Boost" for the combination of over/under-sampling and jittering.

We note that for our procedure we only add noise to the values in the training data for which over-sampling creates replication. For instance, to estimate the $q = 0.9$ quantile in the circle example, the augmented data contains nine instances for every observation from the original training data for which $y = -1$. To eight of these nine we add the *iid* noise and leave the ninth unchanged. No noise is added to the observations for which $y = +1$, as none are replicated in the augmented data. In this way the augmented data contains exactly one copy of the original training data.

The third panel in Figure 12 shows the estimate of $q = 0.9$ for this same simulation after $m = 5000$ iterations with the noise added as described using $\nu = 1/4$. The algorithm still appears to be estimating the $q = 0.9$ quantile even with this large number of iterations and even with the training error on the augmented data set having been zero for the last 1000 of the 5000 total iterations. Note that while the training data does contain a number of $+1$'s outside this grey circle (left panel of Figure 1), the interpolation on these points is so extremely local that it affects only a very few points in our hold-out sample. This gives insight into why boosting classification trees can be an effective classifier in noisy environments despite fitting all the training data completely. From a probabilistic point of view, the training data has zero measure with respect to the population. More discussion of this phenomenon can be found in Mease and Wyner (2007).

4. From a *q*-Classifier to a Probability Estimator

We have discussed and illustrated how a median finder like AdaBoost can be converted to a q-classifier by under-sampling or over-sampling and jittering the data. However, if estimation of the CCPF is the end goal, it is still necessary to further convert the q-classifier to an estimator of the conditional class probabilities. We next present a simple general algorithm to do just that. That is, the following procedure can be used to produce an actual estimate $\hat{p}(x)$ for the CCPF $p(x)$ from a range of q-classifiers.

First a quantization level $\delta > 2$ is fixed. Next, q-classification is carried out on the data for a range of quantiles $q = 1/\delta, 2/\delta, ..., 1 - 1/\delta$. For each q and every x this provides an estimate of $\mathrm{I}\{p(x) \geq q\}$ which we will denote as $\hat{D}_q(x) \in \{0,1\}$. While it is clear that $\mathrm{I}\{p(x) \geq q\}$ is monotonically decreasing in q, the estimates $\hat{D}_q(x)$ will not generally be monotonic, especially when $p(x)$ is close to q. Thus, to produce an estimate of $p(x)$ from the sequence of quantile-based estimates $\hat{D}_q(x)$, monotonicity of the level sets must be achieved by brute force. One solution is to begin with the median $\hat{D}_{0.5}(x)$. The algorithm is as follows.

- If $\hat{D}_{0.5}(x) = 1$ then let $\hat{p}(x) = \min\{q > 0.5 : \hat{D}_q(x) = 0\} - \frac{1}{2\delta}$. If no such q is found take $\hat{p}(x) = 1 - \frac{1}{2\delta}$.
- If $\hat{D}_{0.5}(x) = 0$ then let $\hat{p}(x) = \max\{q < 0.5 : \hat{D}_q(x) = 1\} + \frac{1}{2\delta}$. If no such q is found take $\hat{p}(x) = \frac{1}{2\delta}$.

5. Performance Comparisons

In this section we seek to quantitatively compare the CCPF estimation performance of JOUS-Boost to some competing methods. The competing methods we will consider are LogitBoost, P-AdaBoost, nearest neighbor 10, Random Forests and classification trees from Rpart. We will compare performance on the simulated data sets presented earlier as well as four real data sets taken from the UCI ML Database.

We note that the purpose of these comparisons is not to argue that we have derived a superior method of estimating probabilities/quantiles using boosting. Rather, as discussed in Section 1.2, our purpose is to illustrate that the overfitting of probability/quantile estimates for regular, off-the-shelf boosting is not a problem inherent in boosting, but rather a consequence of believing that applying a link function to the scores should be used to recover probabilities. Most of the difficulty with probability/quantile estimation is solved by simply adopting the more modest view of boosting as a classifier and relinquishing the view of boosting as a probability estimator. Our experiments reveal that simple over- or under- sampling (which is the basis for our algorithm) solves the probability/quantile overfitting problems and produces a competitive algorithm. If we were instead to make an argument for the superiority of the algorithm, there are a large number of variations of boosting, some of which were mentioned in Section 1, which we would need to consider as competitors. These include direct cost-optimization boosting methods as well as regularizing boosting by slowing the learning rate or restricting the flexibility of the weak learners as in the case of decision stumps mentioned earlier.

5.1 Implementation Details

We will use the algorithm in Section 4 to produce an estimate of the CCPF from JOUS-Boost. Specifically, we will apply this algorithm with $\delta = 10$. Consequently our estimate $\hat{p}(x)$ for $p(x)$ at any x will be one of $\{0.05, 0.15, ..., 0.95\}$. This will of course involve performing AdaBoost on nine artificial training data sets. For under-sampling we will create these data sets for each value of q by sampling without replacement from the original training data using $k_{+1} = (1-q)N_{+1}$ and $k_{-1} = qN_{-1}$ (rounded appropriately). The one exception is $q = 0.5$ for which we simply leave the data unmodified, that is, $k_{+1} = N_{+1}$ and $k_{-1} = N_{-1}$. For over-sampling we will create these data sets by replicating observations in the training data using $k_{+1} = \delta(1-q)N_{+1}$ and $k_{-1} = \delta q N_{-1}$, except again in the case where $q = 0.5$ for which we use the original training data. For over-sampling we will add noise as described earlier to any artificially replicated values. Note that one attractive feature of this scheme is that for classification at the 1/2 quantile, the resulting probabilities by construction give rise to a classification rule equivalent to that of AdaBoost on the original data.

To minimize disagreement between the nine quantile estimates we will restrict the sampling such that any (artificial) data set with a smaller value of N_{+1} than that of a second (artificial) data set must not contain any observations for $y = +1$ which are not also in the second data set. The same is true for N_{-1} and $y = -1$. For example, the observations with $y = -1$ in the data set for estimating $q = 0.6$ are a proper subset of those in the data set for $q = 0.7$. Also for the purposes of minimizing the disagreement across the different values of q, when over-sampling we will "recycle" the noise across the different values of q; any observation that appears with noise added to it in more than one of these artificial data sets will have the same value for the noise added to it in all data sets.

The tuning parameter ν will be fixed at a value of 1 throughout the remainder of this article. We have found this to be a fairly "medium" noise level. For the data sets we considered for which the misclassification error and Bayes error are small, values of ν smaller than 1 work better. Likewise, for the data sets for which the misclassification error or Bayes error are larger, values of ν larger than 1 work better. We choose not to tune this parameter so as to provide for a more fair comparison with P-AdaBoost and LogitBoost; however, in practice tuning this parameter through cross-validation would be desirable.

5.2 Simulated Data Sets

We begin by examining the performance for the five simulated data sets presented earlier. These are the four simulations based on the 10-Norm model, first introduced by Friedman et al. (2000) and defined in (2), and the 1000-dimensional simulation defined in (6). For each of these the over-sampling and under-sampling algorithms using AdaBoost as described above are shown on the corresponding plots in Section 2 by the red and green curves respectively.

The original simulation by Friedman et al. (2000) used $n = 5000$ and $\gamma = 10$. The plots in Figure 8 display the misclassification error and our probability scoring functions averaged over the hold-out sample. The under-sampled AdaBoost algorithm performs better than LogitBoost, which exhibits overfitting around 200 iterations, but it does not perform as well as P-AdaBoost. The over-sampled AdaBoost algorithm has the worst performance of the four; however, if the noise parameter ν is set at a smaller value, its performance in this simulation is superior to that of the other three. Again, the amount of noise ν should be regarded as a tuning parameter, but we have kept it fixed throughout these experiments. Note that the fact that P-AdaBoost does not suffer from overfitting in this simulation can be attributed to the low Bayes error, large sample size and moderate number of dimensions as mentioned earlier.

In Figure 9 we saw that when γ was decreased to 0.5 the LogitBoost algorithm severely overfit the probabilities, but P-AdaBoost did not. The over- and under-sampled AdaBoost algorithms perform well in this simulation with the under-sampled AdaBoost algorithm being comparable to P-AdaBoost and over-sampling only slightly worse. Again, by using a smaller value for ν, the over-sampled AdaBoost can be tuned to outperform the others in this simulation.

When the sample size is decreased from 5000 to 500 for this model, Figures 5 and 6 show that both P-AdaBoost and LogitBoost overfit the probabilities more and more as the iterations are increased. However, the under- and over-sampled AdaBoost algorithms actually appear to asymptote to their minima. They provide stable probability estimation regardless of the number of iterations, mimicking the behavior of AdaBoost with respect to misclassification error as intended. Similar behavior can be observed in the plots in Figure 10 for the 1000-dimensional example with respect to under-sampling. Over-sampling was not performed for this simulation due to computational limitations.

Tables 1-4 display misclassification error, squared loss, log loss and exponential loss for P-AdaBoost and LogitBoost as well as the over- and under-sampled AdaBoost algorithms at $m = 800$ iterations. These tables also give the values for other competing methods for CCPF estimation: CART, Random Forests and nearest neighbor 10. The implementation of CART is that of the default classification tree grown by the function Rpart in R. Random Forests are also implemented in R using the default settings. The Bayes column for misclassification gives the error for the hold-out sample using the true (unknown) Bayes rule. All results are based on the same training and test sets in each case. With P-AdaBoost and LogitBoost we continue to threshold all probability estimates at 0.05 and 0.95 as before when computing log loss and exponential loss (but not squared loss). This prevents the over- and under-sampled AdaBoost algorithms from gaining an unfair advantage against overfitting due to their course quantization. We do not include JOUS-Boost in Table 1 because the classification rules for JOUS-Boost are equivalent to AdaBoost by definition.

From these tables it can be seen that the over-sampled and under-sampled AdaBoost algorithms provide a competitive method for probability estimation for these simulations. Their results are comparable to that of Random Forests and nearest neighbor 10 and superior to CART. They do not suffer from the severe overfitting of P-AdaBoost and LogitBoost. Further, even better performance may be achieved for over-sampling by allowing tuning of the parameter ν which controls the amount of noise added to the replicates.

5.3 Real Data Sets

We now consider the two real data sets "Sonar" and "Satimage". Both of these are from the UCI ML database and are used in the article by Friedman et al. (2000).

When using real data sets we can no longer use our scoring rules on the probabilities since the true CCPF is unknown. Instead, we will substitute the value for the true class probabilities in

Simulation Name	AdaBoost	Logit-Boost	CART	Random Forests	Nearest Neighbor 10	Bayes
10-Norm(n=500, γ=10)	0.23	0.25	0.39	0.27	0.28	0.03
10-Norm(n=500, γ=0.5)	0.34	0.36	0.38	0.36	0.37	0.24
10-Norm(n=5000, γ=10)	0.10	0.12	0.31	0.16	0.19	0.03
10-Norm(n=5000, γ=0.5)	0.28	0.29	0.40	0.27	0.30	0.22
1000 Dimensions	0.34	0.35	0.50	0.37	0.44	0.03
Sonar	0.11	0.16	0.26	0.15	0.31	-
Sonar w/ Noise	0.22	0.27	0.35	0.21	0.36	-
Satimage	0.06	0.06	0.12	0.06	0.08	-
Small Satimage w/ Noise	0.21	0.20	0.21	0.18	0.18	-
German Credit	0.25	0.25	0.27	0.23	-	-
Connect 4	0.22	0.22	0.34	0.24	-	-

Table 1: Misclassification Error Comparison

Simulation Name	JOUS (Over-Sampling)	JOUS (Under-Sampling)	P-AdaBoost	Logit-Boost	CART	Random Forests	Nearest Neighbor 10
10-Norm(n=500, γ=10)	0.14	0.15	0.20	0.22	0.25	0.17	0.16
10-Norm(n=500, γ=0.5)	0.07	0.07	0.19	0.19	0.12	0.06	0.07
10-Norm(n=5000, γ=10)	0.09	0.06	0.05	0.07	0.19	0.11	0.12
10-Norm(n=5000, γ=0.5)	0.03	0.03	0.03	0.10	0.09	0.03	0.05
1000 Dimensions	-	0.18	0.27	0.28	0.23	0.21	0.22
Sonar	0.08	-	0.11	0.16	0.21	0.13	0.20
Sonar w/ Noise	0.15	-	0.22	0.26	0.28	0.17	0.21
Satimage	0.06	0.08	0.06	0.05	0.10	0.05	0.05
Small Satimage w/ Noise	0.15	0.16	0.19	0.20	0.17	0.14	0.15
German Credit	-	0.18	0.24	0.22	0.20	0.16	-
Connect 4	-	0.20	0.21	0.21	0.23	0.16	-

Table 2: Squared Loss Comparison

Simulation Name	JOUS (Over-Sampling)	JOUS (Under-Sampling)	P-AdaBoost	Logit-Boost	CART	Random Forests	Nearest Neighbor 10
10-Norm(n=500, γ=10)	0.49	0.50	0.69	0.75	0.76	0.55	0.54
10-Norm(n=500, γ=0.5)	0.63	0.63	1.04	1.06	0.77	0.61	0.63
10-Norm(n=5000, γ=10)	0.38	0.29	0.25	0.33	0.61	0.43	0.43
10-Norm(n=5000, γ=0.5)	0.53	0.52	0.52	0.77	0.66	0.54	0.57
1000 Dimensions	-	0.62	0.93	0.94	0.71	0.68	0.70
Sonar	0.26	-	0.36	0.51	0.65	0.41	0.56
Sonar w/ Noise	0.48	-	0.70	0.80	0.86	0.51	0.60
Satimage	0.21	0.29	0.22	0.21	0.35	0.19	0.19
Small Satimage w/ Noise	0.51	0.49	0.60	0.63	0.52	0.46	0.46
German Credit	-	0.55	0.75	0.68	0.59	0.49	-
Connect 4	-	0.59	0.68	0.65	0.65	0.49	-

Table 3: Log Loss Comparison

Simulation Name	JOUS (Over-Sampling)	JOUS (Under-Sampling)	P-AdaBoost	Logit-Boost	CART	Random Forests	Nearest Neighbor 10
10-Norm(n=500, γ=10)	0.78	0.79	1.12	1.20	1.10	0.85	0.84
10-Norm(n=500, γ=0.5)	0.95	0.94	1.61	1.63	1.12	0.92	0.93
10-Norm(n=5000, γ=10)	0.65	0.53	0.48	0.61	0.92	0.71	0.71
10-Norm(n=5000, γ=0.5)	0.82	0.80	0.80	1.20	0.97	0.83	0.86
1000 Dimensions	-	0.92	1.43	1.43	1.02	0.99	1.01
Sonar	0.50	-	0.67	0.88	1.00	0.69	0.85
Sonar w/ Noise	0.79	-	1.14	1.26	1.28	0.81	0.90
Satimage	0.43	0.53	0.46	0.45	0.62	0.41	0.40
Small Satimage w/ Noise	0.85	0.79	0.99	1.03	0.82	0.76	0.77
German Credit	-	0.86	1.19	1.08	0.90	0.78	-
Connect 4	-	0.89	1.11	1.06	0.96	0.78	-

Table 4: Exponential Loss Comparison

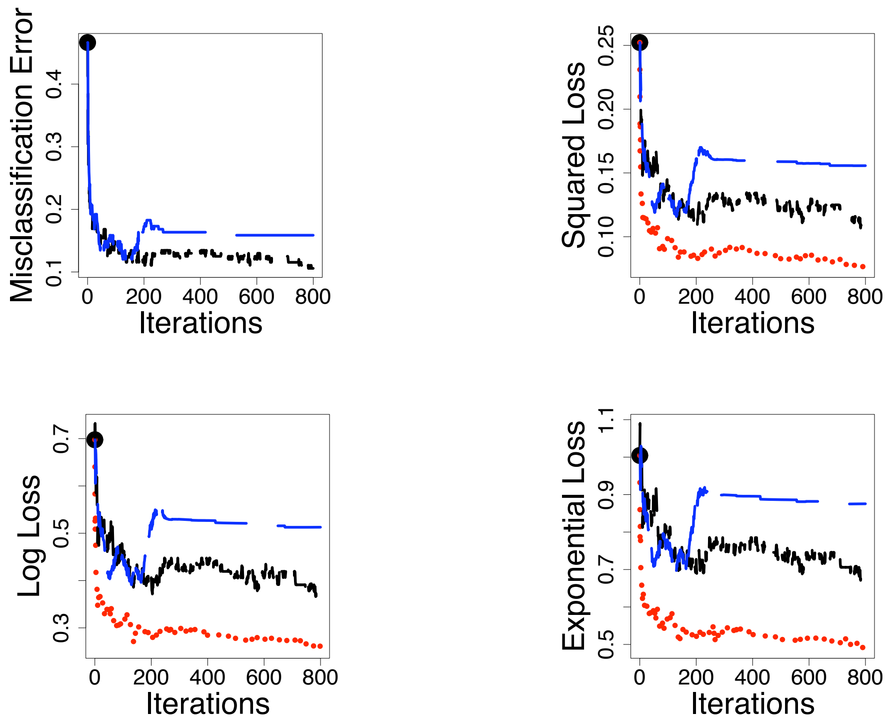


Figure 13: Misclassification error, squared loss, log loss and exponential loss for Sonar. **Black (Short Dashes)**: P-AdaBoost. **Blue (Long Dashes)**: LogitBoost. **Red (Dotted)**: JOUS-Boost, over-sampled.

the hold-out samples with the indicator of whether or not the observed class is one. That is, we substitute the true probability function $p(x) = \mathrm{P}[Y = 1|x]$, which is unknown by $1\{y = 1\}$, which is observed. With this substitution we can evaluate the loss functions (Equations (3), (4) and (5)) as before. Note that since all three of the scoring rules we are using are *proper scoring rules* (Savage, 1973) they are minimized in expectation as a function of $\hat{p}$ by the true probability p even with this substitution of the indicator for the probability.

The Sonar data set has 208 observations, $d = 60$ predictors and two classes. Since there is no designated test sample, 10-fold cross validation was used. While Figure 13 reveals both P-AdaBoost and LogitBoost have a general decreasing trend with respect to the probability scoring rules (excepting the strange jump by LogitBoost right before $m = 200$), both algorithms clearly could benefit from not estimating all probabilities with (essentially) zero and one as evidenced by the superior performance of the over-sampled AdaBoost algorithm. The fact that both LogitBoost and P-AdaBoost estimate all probabilities with values close to zero or one is reflected in the similarity of their respective values for squared loss and misclassification error. We did not apply under-sampling on this data set due to the small number of observations.

The negative effects of overfitting probabilities with P-AdaBoost and LogitBoost are amplified if we consider artificially adding "label noise." By label noise we mean that with some probability

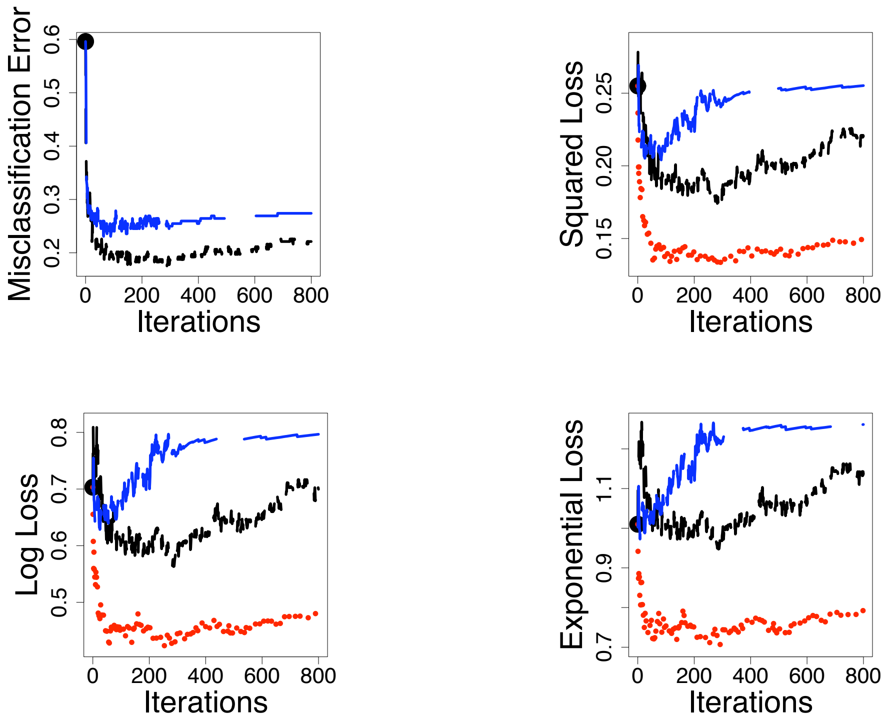


Figure 14: Misclassification error, squared loss, log loss and exponential loss for Sonar with noise. **Black (Short Dashes)**: P-AdaBoost. **Blue (Long Dashes)**: LogitBoost. **Red (Dotted)**: JOUS-Boost, over-sampled.

q we randomly change the signs of the class labels in the data set. The plots in Figure 14 show the results for label noise of $q = 0.1$ over the same ten fold resamples.

The Satimage data set has a designed training set with $n = 4435$ observations and a hold-out set of $n^* = 2000$ observations. There are $d = 36$ predictors. The data set has six classes labelled as $1, 2, 3, 4, 5$ and 7. Since we are only dealing with two-class classification for the purposes of this article we will take one class to be 1 and 7 and the other class to be 2, 3, 4 and 5.

The four plots in Figure 15 show misclassification error, squared loss, log loss and exponential loss averaged over the designated hold-out sample of $n^* = 2000$ for over- and under- sampled AdaBoost, LogitBoost and P-AdaBoost. Neither P-AdaBoost nor LogitBoost seem to increase as a result of overfitting probabilities and perform similarly. Over-sampled AdaBoost performs only slightly better than both P-AdaBoost and LogitBoost, while under-sampled AdaBoost is the worst of the four, although it is still decreasing somewhat quickly at $m = 800$.

While P-AdaBoost and LogitBoost seem to perform well for probability estimation on the Satimage data set, the negative effects of the overfitting can be seen by again adding 10% label noise. Adding this noise severely diminishes the probability estimation of LogitBoost, although P-AdaBoost still performs well (not shown). To see the impact of overfitting for P-AdaBoost in this noisy environment it is sufficient to take a random sample of 1000 from the original 4435 ("Small Satimage"). The corresponding plots are given in Figure 16. Both under-sampling and

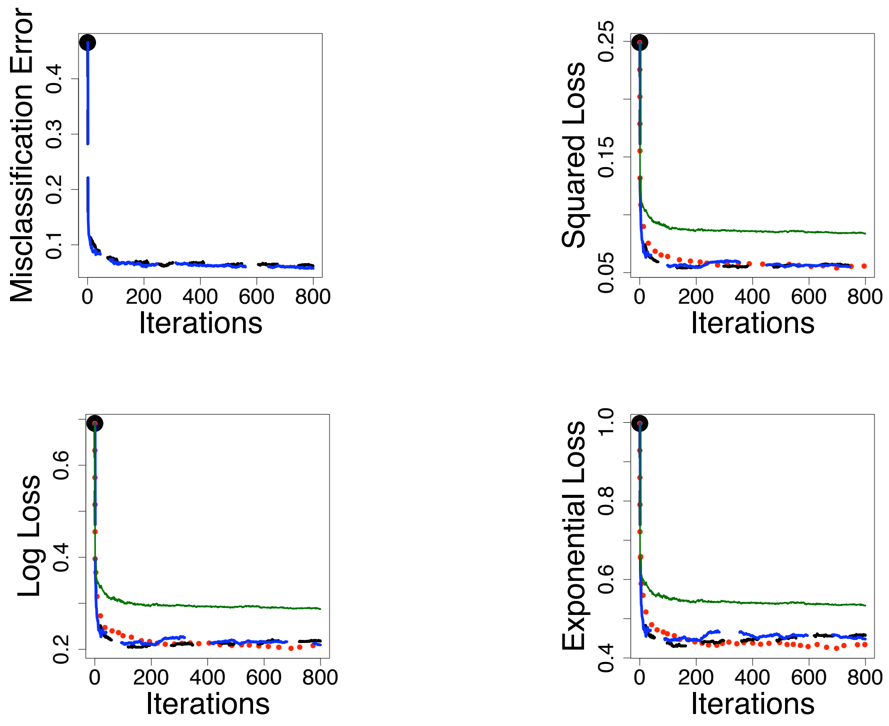


Figure 15: Misclassification error, squared loss, log loss and exponential loss for Satimage. **Black (Short Dashes)**: P-AdaBoost. **Blue (Long Dashes)**: LogitBoost. **Red (Dotted)**: JOUS-Boost, over-sampled. **Green (Solid)**: JOUS-Boost, under-sampled.

over-sampling continue to perform well on this smaller data set, with under-sampling now being superior to over-sampling with respect to log loss and exponential loss.

Tables 1-4 also give results at $m = 800$ iterations for the Sonar and Satimage data sets as well as the modified versions considered. The JOUS-Boost algorithm outperforms CART, Random Forests and nearest neighbor 10 on Sonar and is competitive for Satimage.

5.4 Data Sets with Categorical Predictors

Up until this point, we have considered data sets with only continuous predictors. To illustrate that the overfitting in terms of CCPF estimation with P-AdaBoost and LogitBoost can still occur with categorical predictors, we now turn our attention to two such data sets from the UCI ML database: "German Credit" and "Connect 4". The German Credit data set contains 1000 observations with 20 predictors, 13 of which are categorical and 7 of which are continuous. The Connect 4 data set contains 67,557 observations with 42 predictors, all of which are categorical. This data set has three classes so we merged the two classes "loss" and "draw" into a single class.

The results for these two data sets are displayed in Figures 17 and 18. The final results at $m = 800$ iterations are also included in Tables 1-4. For German Credit, 10-fold cross validation was used, while for Connect 4, the data was randomly split into 40,000 training observations and 27,557

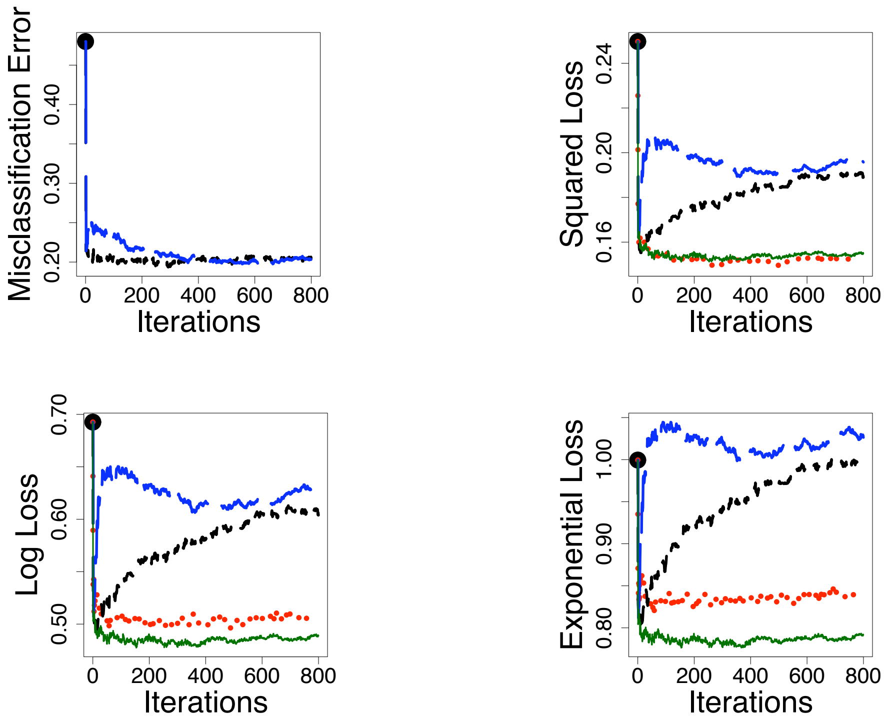


Figure 16: Misclassification error, squared loss, log loss and exponential loss for Small Satimage with noise. **Black (Short Dashes)**: P-AdaBoost. **Blue (Long Dashes)**: LogitBoost. **Red (Dotted)**: JOUS-Boost, over-sampled. **Green (Solid)**: JOUS-Boost, under-sampled.

test observations. Also, due to the large size of this data set, we used 2^{10}-node trees instead of 8-node trees, as this gave superior classification performance. We considered only under-sampling for these two data sets. Over-sampling was not used because it is not obvious how to add noise to categorical predictors.

As can be seen in Figures 17 and 18, both LogitBoost and P-AdaBoost exhibit substantial overfitting of the CCPF on these two data sets, while under-sampled AdaBoost shows only a relatively small amount of overfitting for German Credit and no overfitting for Connect 4. In terms of performance, Tables 1-4 show that under-sampled AdaBoost outperforms CART for both data sets, but that Random Forests is most effective overall.

6. Concluding Remarks

Boosting algorithms such as AdaBoost are known to perform well for classification and are very resistant to overfitting with respect to misclassification error, even though conditional class probability estimates eventually diverge to zero and one, implying complete overfit in terms of CCPF estimation but not classification. With Friedman et al.'s (2000) analysis of boosting in mind, we allow that one may still choose to interpret boosting classification trees as additive logistic regression,

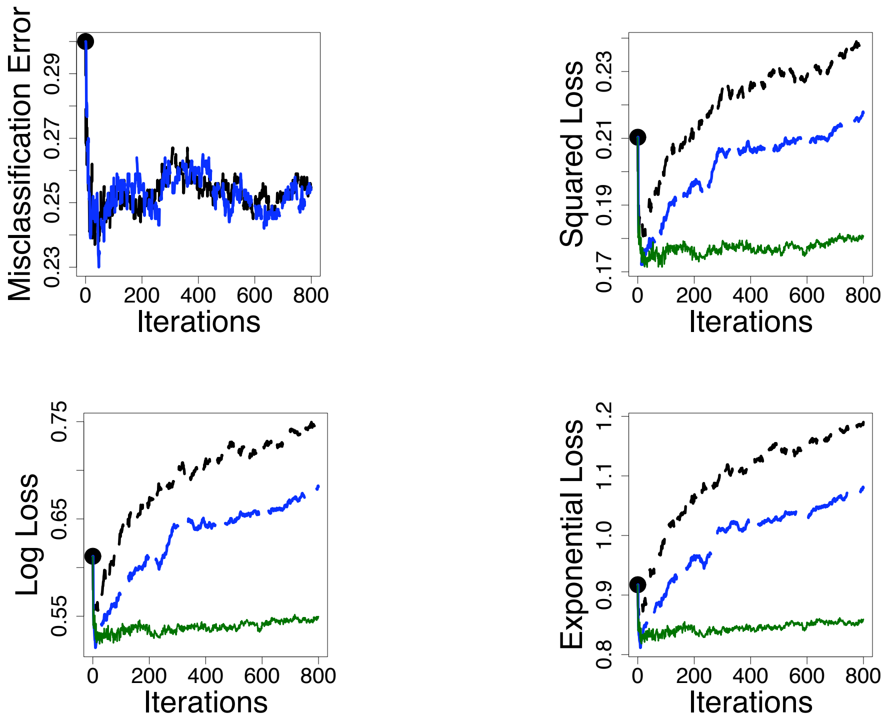


Figure 17: Misclassification error, squared loss, log loss and exponential loss for German Credit. **Black (Short Dashes)**: P-AdaBoost. **Blue (Long Dashes)**: LogitBoost. **Green (Solid)**: JOUS-Boost, under-sampled.

with the understanding, however, that the probabilities may need to be drastically overfit to obtain optimal classification. That is, when attempting classification, one may lose the connection with conditional class probability estimation.

There remains the practical problem of classification in the presence of unequal misclassification costs, or equivalently classification at quantiles other than 1/2, or yet equivalently classification for future base rates that are different from those in the training sample. Viewing AdaBoost or LogitBoost as a form of logistic regression may suggest a *practice* that often leads to poor performance. If one seeks a quantile other than the median and adopts this view of boosting algorithms, then one must hope to stop the boosting process early enough that the absolute value of the score function is not too large relative to the link function evaluated at the quantile of interest, but at the same time late enough that the boosting process has had ample time to learn the truth. Even if such a stopping time exists (we have shown examples where it does not), it may be difficult to find.

In contrast, the modest view that boosting produces only median classifiers suggests more effective modifications to the algorithm in order to obtain quantiles other than the median. One such modification is based on over/under-sampling with a fix (jittering) to eliminate problems with ties (here called JOUS-Boost). We have indicated that JOUS-Boost is often more successful through a number of examples, including some used by Friedman et al. (2000).

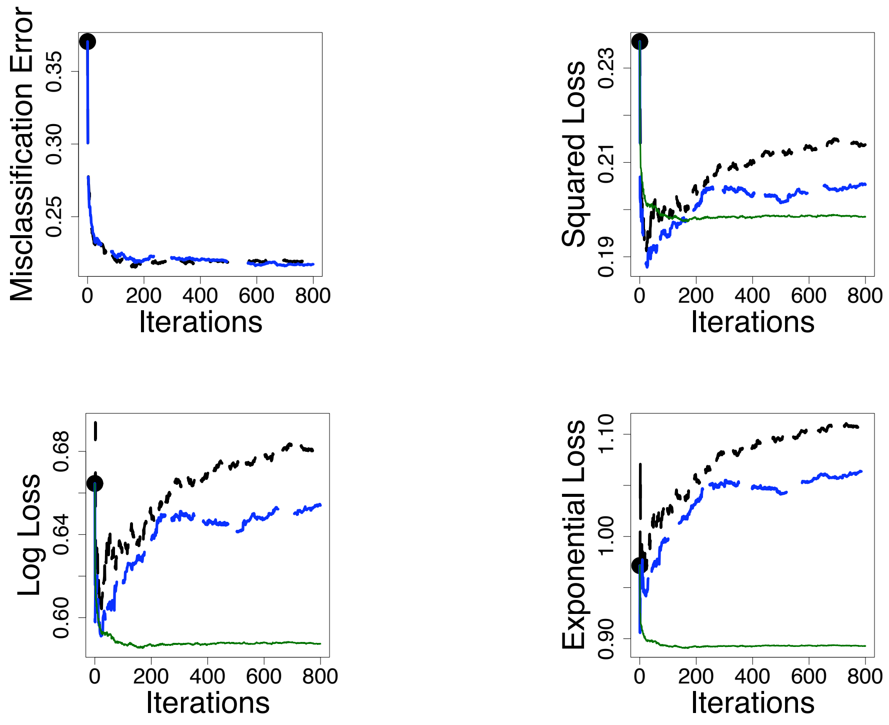


Figure 18: Misclassification error, squared loss, log loss and exponential loss for Connect 4. **Black (Short Dashes)**: P-AdaBoost. **Blue (Long Dashes)**: LogitBoost. **Green (Solid)**: JOUS-Boost, under-sampled.

Finally, we have demonstrated that JOUS-Boost classifiers obtained for a grid of quantiles can be combined to create class probability estimators that perform well with respect to probability estimation.

Acknowledgments

David Mease's work was supported by an NSF-DMS post-doctoral fellowship. We are grateful to Abba Krieger and Dean Foster for helpful comments and suggestions.

References

P. J. Bickel, Y. Ritov, and A. Zakai. Some theory for generalized boosting algorithms. *Journal of Machine Learning Research*, 7:705–732, 2006.

G. Blanchard, G. Lugosi, and N. Vayatis. On the rate of convergence of regularized boosting classifiers. *Journal of Machine Learning Research*, 4:861–894, 2003.

A. Buja, W. Stuetzle, and Y. Shen. Loss functions for binary class probability estimation and classification: Structure and applications. 2006.

P. Chan and S. Stolfo. Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 164–168, 1998.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer. Smoteboost: Improving prediction of the minority class in boosting. In *Proceedings of 7th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 107–119, 2003.

W. W. Cohen and Y. Singer. A simple, fast, and effective rule learner. In *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI)*, pages 335–342, 1999.

M. Collins, R. E. Schapire, and Y. Singer. Logistic regression, adaboost and bregman distances. In *Computational Learing Theory*, pages 158–169, 2000.

M. Dettling and P. Buhlmann. Boosting for tumor classification with gene expression data. *Bioinformatics*, 19:1061–1069, 2003.

N. Duffy and D. Helmbold. Potential boosters? In *Advances in Neural Information Processing Systems*, pages 258–264, 1999.

C. Elkan. The foundations of cost-sensitive learning. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 973–978, 2001.

A. Estabrooks, T. Jo, and N. Japkowicz. A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, 20:18–36, 2004.

W. Fan, S. Stolfo, J. Zhang, and P. Chan. Adacost: Misclassification cost-sensitive boosting. In *Proceedings of the 16th International Conference on Machine Learning*, pages 97–105, 1999.

D. P. Foster and R. A. Stine. Variable selection in data mining: Building a predictive model for bankruptcy. *Journal of the American Statistical Association*, 99:303–313, 2004.

Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Machine Learning: Proceedings of the Thirteenth International Conference*, pages 148–156, 1996.

Y. Freund and R. E. Schapire. Discussion of three papers regarding the asymptotic consistency of boosting. *Annals of Statistics*, 32:113–117, 2004.

J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:337–374, 2000.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

W. Jiang. Process consistency for adaboost. *Annals of Statistics*, 32:13–29, 2004.

W. Jiang. Does boosting overfit: Views from an exact solution. *Technical Report 00-03, Department of Statistics, Northwestern University*, 2000.

M. Joshi, V. Kumar, and R. Agarwal. Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In *Proceedings of the First IEEE International Conference on Data Mining (ICDM)*, pages 257–264, 2001.

G. Lebanon and J. Lafferty. Boosting and maximum likelihood for exponential models. In *Advances in Neural Information Processing Systems*, 2001.

Y. Liu, Y. Yang, and J. Carbonell. Boosting to correct inductive bias in text classification. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, pages 348–355, 2002.

G. Lugosi and N. Vayatis. On the bayes-risk consistency of regularized boosting methods. *Annals of Statistics*, 32:30–55, 2004.

D. Mease and A. Wyner. Evidence contrary to the statistical view of boosting. 2007.

S. Rosset, J. Zhu, and T. Hastie. Boosting as a regularized path to a maximum margin classifier. *Journal of Machine Learning Research*, 5:941–973, 2004.

L. J. Savage. Elicitation of personal probabilities and expectations. *Journal of the American Statistical Association*, 66:783–801, 1973.

K. M. Ting. A comparative study of cost-sensitive boosting algorithms. In *Proceedings of the 17th International Conference on Machine Learning*, pages 983–990, 2000.

B. Zadrozny and C. Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 609–616, 2001.

T. Zhang and B. Yu. Boosting with early stopping: Convergence and consistency. *Annals of Statistics*, 33:1538–1579, 2005.

Journal of Machine Learning Research 8 (2007) 441-479 Submitted 3/06; Revised 10/06; Published 3/07

Value Regularization and Fenchel Duality

Ryan M. Rifkin RRIFKIN@HONDA-RI.COM
Honda Research Institute USA, Inc.
One Cambridge Center, Suite 401
Cambridge, MA 02142, USA

Ross A. Lippert LIPPERT@MATH.MIT.EDU
Department of Mathematics
Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139-4307, USA

Editor: Peter Bartlett

Abstract

Regularization is an approach to function learning that balances fit and smoothness. In practice, we search for a function f with a finite representation $f = \sum_i c_i \phi_i(\cdot)$. In most treatments, the c_i are the primary objects of study. We consider *value regularization*, constructing optimization problems in which the predicted values at the training points are the primary variables, and therefore the central objects of study. Although this is a simple change, it has profound consequences. From convex conjugacy and the theory of Fenchel duality, we derive separate optimality conditions for the regularization and loss portions of the learning problem; this technique yields clean and short derivations of standard algorithms. This framework is ideally suited to studying many other phenomena at the intersection of learning theory and optimization. We obtain a value-based variant of the representer theorem, which underscores the transductive nature of regularization in reproducing kernel Hilbert spaces. We unify and extend previous results on learning kernel functions, with very simple proofs. We analyze the use of unregularized bias terms in optimization problems, and low-rank approximations to kernel matrices, obtaining new results in these areas. In summary, the combination of value regularization and Fenchel duality are valuable tools for studying the optimization problems in machine learning.

Keywords: kernel machines, duality, optimization, convex analysis, kernel learning

1. Introduction

Given a set of training data $\{(X_1,Y_1),\dots,(X_n,Y_n)\}$, the inductive supervised learning task is to learn a function f that, given a new X value, will predict the associated Y value. A common framework for solving this problem is Tikhonov regularization (Tikhonov and Arsenin, 1977):

$$\inf_{f\in\mathcal{F}} \left\{ \sum_{i=1}^{n} v(f(X_i),Y_i) + \frac{\lambda}{2}\Omega(f) \right\}. \tag{1}$$

In this general form, $\mathcal{F}$ is a space of functions from which f must be selected, and $v(f(X_i),Y_i)$ is the *loss*, indicating the penalty we pay when we see X_i, predict $f(X_i)$, and the true value is Y_i. For a large class of functions $\mathcal{F}$, simply minimizing $\sum_{i=1}^{n} v(f(X_i),Y_i)$ directly is ill-posed and leads to overfitting the training data. We restore well-posedness by introducing a *regularization* term $\Omega(f)$

that penalizes elements of f that are too "complex". The regularization parameter, λ, controls the tradeoff between finding a function of low complexity and fitting the training data.

A large amount of work (Wahba, 1990; Evgeniou et al., 2000) takes $\mathcal{F}$ to be a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ (Aronszajn, 1950) induced by a kernel function k. In this situation, we rewrite (1) as

$$\inf_{f\in\mathcal{H}} \left\{ \sum_{i=1}^{n} v(f(X_i),Y_i) + \frac{\lambda}{2}||f||_k^2 \right\}, \tag{2}$$

indicating that the regularization term is now the squared norm of the function in the RKHS. Many common algorithms, including support vector machines for classification (Cortes and Vapnik, 1995) and regression (Vapnik, 1998), regularized least squares (Wahba, 1990; Poggio and Girosi, 1990; Rifkin, 2002), and kernel logistic regression (Jaakkola and Haussler, 1999) are instances of Tikhonov regularization: different loss functions yield different algorithms.[1]

A consequence of the so-called *representer theorem* (Wahba, 1990; Schölkopf et al., 2001), is that the minimizer of (2) will have the form

$$f(x) = \sum_{i=1}^{n} c_i k(x, X_i). \tag{3}$$

In other words, in an RKHS, a function f which minimizes (2) is a sum of kernel functions $k(\cdot, X_i)$ over the data points. We solve a Tikhonov regularization (or, equivalently, train a predictor) by finding the coefficients c_i. Assuming that the loss function is convex, the minimization of (2) (or (1)) is tractable.

Defining K as the $n \times n$ *kernel matrix* with $K_{ij} = k(X_i, X_j)$, the regularization penalty $||f||_k^2$ becomes $c^t K c$, the output at training point i is given by $f(X_i) = \sum_{j=1}^{n} c_j k(X_i, X_j) = (Kc)_i$, and we can rewrite (2) as

$$\inf_{c\in\mathbb{R}^n} \left\{ \sum_{i=1}^{n} v((Kc)_i, Y_i) + \frac{\lambda}{2} c^t K c \right\}. \tag{4}$$

Regularizing in an RKHS is special in that it has a representer theorem: the optimal solution in an infinite-dimensional space of functions is found by solving a finite dimensional minimization problem. Although most other function space regularizers do not have similar properties, we may consider other regularizers as modifications of (4): replacing $c^t K c$ with $c^t c$, we obtain *ridge regression* (Tikhonov and Arsenin, 1977), and replacing it with $\sum_{i=1}^{n} |c_i|$, we obtain *L1-regularization* (Zhu et al., 2003). In such cases, we are assuming that we are looking for a function of the form (3) a priori. There is nothing wrong with this, but it should not be confused with the case of RKHS regularization, where (3) arises naturally from a search for an optimizing function.

In general, an equation like (4) is used as the starting point for thinking algorithmically about finding c. For example, in a standard development of support vector machines (Cristianini and Shaw-Taylor, 2000; Rifkin, 2002), one starts with (4) instantiated with the SVM *hinge loss* $v(f(X_i),Y_i) = (1 - y_i f(X_i))_+$, introducing slack variables $\xi_i = v(f(X_i),Y_i)$ and constraints to handle

1. Technically speaking, to derive an algorithm such as the classic SVM one also needs an *unregularized* bias term b: this issue is discussed in detail later in the paper.

the non-differentiability of the hinge loss at 0 as well as an unregularized bias term b, yielding a quadratic program:[2]

$$\begin{array}{lll} \min\limits_{c\in\mathbb{R}^n,\xi\in\mathbb{R}^n,b\in\mathbb{R}} & \frac{1}{n}\sum_{i=1}^n \xi_i + \lambda c^T K c & \\ \text{subject to :} & Y_i\left(\sum_{j=1}^n c_j k(X_i,X_j)+b\right) \geq 1-\xi_i & i=1,\ldots,n \\ & \xi_i \geq 0 & i=1,\ldots,n. \end{array}$$

This program is called the *primal problem*. In order to expose sparsity in the solution, the Lagrangian dual is taken, yielding the *dual problem*:

$$\begin{array}{lll} \max\limits_{\alpha\in\mathbb{R}^n} & \sum_{i=1}^n \alpha_i - \frac{1}{(2\lambda)^2}\alpha^t \operatorname{diag}(Y) K \operatorname{diag}(Y)\alpha & \\ \text{subject to :} & \sum_{i=1}^n Y_i\alpha_i = 0 & \\ & 0 \leq \alpha_i \leq \frac{1}{n} & i=1,\ldots,n. \end{array}$$

It is then observed that the dual problem is easier to solve (because of its simpler constraint structure), and that solutions to the primal can be easily obtained from solutions to the dual.

We propose to take a different approach to Tikhonov regularization, that we believe to be more fundamental. The approach rests on two ideas.

First, we consider the *predicted values* $y_i \equiv f(X_i) = (Kc)_i$ to be the central objects of study, and write our optimization problems in terms of y. In the case of RKHS regularization with a positive-definite kernel, the matrix K is typically non-singular (Micchelli, 1986), [3] and we rewrite (4) as a *value regularization*:

$$\inf_{y\in\mathbb{R}^n}\left\{\frac{\lambda}{2}y^t K^{-1} y + \sum_{i=1}^n v(y_i,Y_i)\right\}.$$

Although this is a simple transformation, the consequences are far-reaching. Looking at the rewritten problem, we notice that the kernel matrix K appears only in the regularization—*it does not appear in the loss term*. This is intuitive, as the loss function (of course) cares only about the predicted outputs, not what combination of kernel coefficients generated those predicted outputs. Additionally, we see that the loss function decomposes into n separate single-point loss functions. In contrast, if the c_i are the primary variables, the loss at each data point is a function of the entire c vector.

The benefits of value regularization are greatly amplified by the second central idea of this work: instead of Lagrangian duality, we use *Fenchel duality* (Borwein and Lewis, 2000), a form of duality that is well-matched to the problems of learning theory. Although we discuss Fenchel duality in greater detail below, we present a brief overview here. Consider an optimization problem of the form:

$$\inf_{y\in\mathbb{R}^n}\{f(y)+g(y)\}. \tag{5}$$

2. Traditional derivations parametrize the loss function instead of the regularization, with a constant $C = \frac{1}{2\lambda}$, but that is a minor point.
3. Throughout this paper, when we work with an RKHS regularizer we will generally assume that the kernel function is positive definite and the points are in general position, implying the existence of K^{-1}. This assumption can be easily relaxed using pseudoinverses, although the mathematics becomes somewhat more cumbersome.

Fenchel duality defines a so-called Fenchel dual:

$$\sup_{z \in \mathbb{R}^n} \{-f^*(z) - g^*(-z)\}, \tag{6}$$

where f^*, g^* are *Fenchel-Legendre conjugates* (definition 4) computed via auxiliary (and by design, simpler) optimization problems on f and g *separately*. Convexity of f and g (and some topological qualifications) ensure that the optimal objective values of (5) and (6) are equivalent, and that any optimal y and z satisfy:

$$\begin{aligned} f(y) - y^t z + f^*(z) &= 0 \\ g(y) + y^t z + g^*(-z) &= 0. \end{aligned}$$

Fenchel duality encompasses other notions of duality such as Lagrangian duality and seems to be a natural concept to apply to regularization problems where f and g each arise from different considerations—for supervised learning problems, one will come from regularization and the other from empirical loss.

Elaborating on this idea, Fenchel duality gives us a *separation of concerns* which is not present in the Lagrangian approach. The point of formulating an optimization problem such as a quadratic program is ultimately to derive optimality conditions and algorithms for finding optimal solutions. For convex optimization problems, all local optima are globally optimal, and we can formulate a complete set of optimality conditions which the primal and dual solutions will simultaneously satisfy. These are generally known as the Karush-Kuhn-Tucker (KKT) conditions (Bazaraa et al., 1993).[4] Fenchel duality makes it clear that the two functions f and g, or, in our case, the loss term and the regularization, contribute *individual* local optimality conditions, and the total optimality conditions for the overall problem are simply the union of the individual optimality conditions. Put differently, using Fenchel duality, we can derive a table for n_r regularizations and n_l losses, and immediately combine these to derive optimality conditions for $n_r n_l$ learning problems. This idea is obscured by the Lagrangian duality approach to deriving optimality conditions for learning problems. As an example, for the common case of the RKHS regularizer $\frac{1}{2} y^t K^{-1} y$, we will find that the primal-dual relationship is given by $y = \lambda^{-1} K z$. This condition is *independent of the loss*—it shows up simultaneously in SVM, regularized least squares, and logistic regression.

Value regularization and Fenchel duality reinforce each other's strengths. Because the kernel affects only the regularization term and not the loss term, applying Fenchel duality to value regularization yields extremely clean formulations. The major contribution of this paper is the combination of value regularization and Fenchel duality in a framework that yields new insights into many of the optimization problems that arise in learning theory.

We present, in Section 3, a primer on convex analysis and Fenchel duality, with emphasis on the key ideas that are needed for learning theory. We believe that this section provides a sufficiently self-contained summary of convex analysis to allow a machine learning researcher to apply our framework to new problems.

In Section 4, we specialize the convex analysis results to Tikhonov value regularizations consisting of the sum of a regularization and a loss term. Section 5 specializes the result further to the RKHS case and obtains very simple derivations of well-known kernel machines.

Section 6 is concerned with value regularization in the context of L1 regularization, a regularization with the explicit goal of obtaining sparsity in the finite representation. In Section 6.1, we

4. The complete KKT conditions for SVM can be found (among other places) in Rifkin (2002).

briefly discuss 1-norm support vector machines, and emphasize a key distinction between RKHS and other regularizations: in RKHS regularization, the c_i, which are the expansion coefficients in the finite representation of the learned function, and the z_i, which are the dual variables associated with the y_i in the value regularization problem, are identifiable ($z = \lambda^{-1}c$). This is a consequence of the RKHS regularization, and does not hold for more general regularizations. In Section 6.2, we present a simpler derivation of the relationship between support vector regression and sparse approximation, first discovered in Girosi (1998); essentially, the problems are duals, with the width of the ε-tube in the support vector regression problem becoming the sparsity regularizer in the sparse approximation problem.

In Section 7, we develop a new view of the representer theorem, in the context of value regularization. The common wisdom about the representer theorem is that it guarantees that the solution to an optimization problem in an (infinite-dimensional) function space has a finite representation as an expansion of kernel functions around the training points. While this is certainly true, we gain additional insights by considering an augmented problem in which the test points also appear in the regularization, but not in the loss. In the augmented optimization problem the predicted outputs at the training points do not change, and the expansion coefficients at the test points vanish—a representer theorem. This underscores the fact that for supervised learning in an RKHS, induction and transduction are identical. In the context of transductive or semi-supervised algorithms, the picture is more complex. There have been several recent articles that turn transductive algorithms into semi-supervised algorithms via an appeal to the representer theorem. While this is formally valid, the resulting transductive algorithm is *not* equivalent to the semi-supervised algorithm, and the transductive algorithm is perhaps the more "natural" choice. Section 7.1 is devoted to a discussion of this topic.

Recently, there has been interest in "learning the kernel". In Section 8, we may view this as a value regularization where the kernel matrix itself is an auxiliary parameter to be optimized. The value-based formulation is ideal here, because the kernel appears only in the regularization term and not in the loss term. We first derive a general result that gives optimality conditions for a general convex penalty $F(K)$ on the kernel matrix. Work to date has considered only the case where F is 0 for some set of semi-definite matrices, and infinite otherwise. Lanckriet et al. (2004) considers a case where the kernel function is a linear combination of a finite set of kernels; we will see that a minor modification to their formulation yields a representer theorem and an agreement of inductive and transductive algorithms. Argyriou et al. (2005) work with a convex set generated by an infinite, continuously parametrized set of kernel functions. We give proofs of their main results which are shorter and simpler, and also more general, allowing arbitrary convex loss functions (Argyriou et al. (2005) requires differentiability).

In Section 9, we show how infimal convolutions, a form of optimization relaxation (see Section 3.4) are useful in learning theory. We first explore the idea of "biased" regularizations, the best-known example being the unregularized bias term b in the standard formulation of support vector machines. We show that unregularized bias terms arise from infimal convolutions and that the optimality conditions implied by bias terms are independent of both the particular loss function and the particular regularization. For example, including an unregularized constant term b in any Tikhonov optimization problem yields a constraint on the dual variables $\sum z_i = 0$; this constraint does not require a particular loss function, or even that we work in an RKHS. More generally, we can include unregularized polynomial functions or even include an unregularized element of an arbitrary convex set. In Section 9.2, we see that the computation of leave-one-out values can also be viewed as a biased regularization via infimal convolution.

In Section 10, we explore low rank kernel matrix approximations such as the Nyström approximation, which consists of picking a partition $N \cup M$ of the data, and approximating K with $\tilde{K} = \begin{pmatrix} K_{NN} & K_{NM} \\ K_{MN} & K_{NM}K_{MM}^{-1}K_{MN} \end{pmatrix}$. In Willams and Seeger (2000), the authors suggest using $\tilde{K}$ in place of K. Several authors (Rifkin, 2002; Rasmusen and Williams, 2006) have found empirically that this works poorly, but that a closely related approach (the *subset of regressors* method) of constraining c_i to be zero for points not in the subset M works quite well. These two approaches are extremely closely related mathematically. With value regularization and Fenchel duality, we explore in detail the relation between these methods. The subset of regressors method is, in some sense, the "natural" algorithm arising from the low-rank matrix approximation, while the Nyström method makes an unwarranted identification of the dual variables, z, and the coefficients of expansion in the finite representation, c.

The framework presented here requires a moderate mathematical investment by the reader. For this reason, we have organized the paper so that the material on convex analysis (Section 3) can be digested in several chunks and give a roadmap (Figure 1) to illustrate which sections are accessible using only a subset of Section 3. Additionally, in Section 3.5, we provide a *cheat sheet* of key ideas from convex analysis, stated without their full qualifying conditions. Alternatively, readers may use the roadmap to skip unnecessary review.

Fenchel duality will find many uses in machine learning theory. Very recently (while the present paper was under review), Dudík and Schapire (2006) used Fenchel duality to explore maximum entropy distribution estimation under constraints, and Altun and Smola (2006) explored relations between divergence minimization and statistical inference. In summary, Fenchel duality is a powerful way to look at the regularization problems that arise in learning theory. While it requires some mathematical sophistication, the resulting formulations are elegant and yield new insights. We hope that many people will find this approach useful.

2. Notation

A training set is a set of labelled points $\{(X_1,Y_1),\ldots,(X_n,Y_n)\}$. We will sometimes use N to refer to the set $\{x_1,\ldots,x_n\}$, and we may have an additional set of (unlabelled) points of size m called M (e.g., $N = \{X_1,\ldots,X_n\}$ and $M = \{X_{n+1},\ldots,X_{n+m}\}$).

Throughout this paper, Y_i (capitalized) refers to given labels for training points. y_i are variables that we optimize over. In general, we can imagine that we are learning a function f, and $y_i = f(X_i)$, but we generally think of optimizing the y_i directly rather than using f as an intermediary.

Beginning in Section 3, we will frequently take Fenchel-Legendre conjugates and use z to denote variables conjugate to y (i.e., we are using z to refer to "dual variables"). We will also sometimes obtain functions of the form $f(x) = \sum_i c_i k(X_i,x)$ and exclusively use c_i to refer to the expansion coefficients in the finite representation of f.

We define e_i to be the n-vector whose ith entry is 1 and whose other entries are zero: the ith basis vector in the standard basis for $\mathbb{R}^n$ (n will always be clear from context). We define 1_n to be a vector of length n whose entries are all 1.

We use H to refer to affine (or hyperplane) functions, $H_{v,c}(y) = v^t y - c$. For any symmetric positive semidefinite matrix, $Q_A(y) = \frac{1}{2}y^t Ay$. We write $(y)_+ = \max\{y,0\}$.

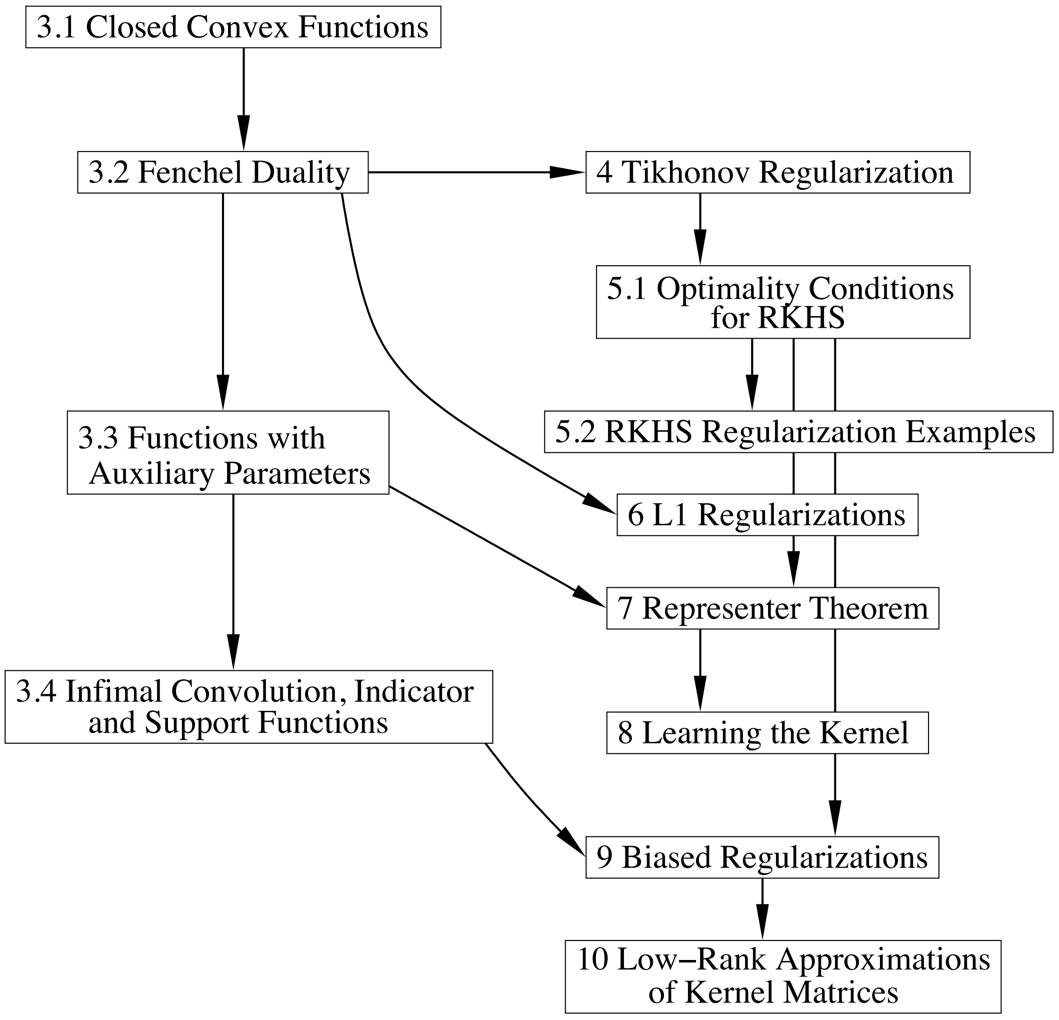

3.1 Closed Convex Functions
3.2 Fenchel Duality
4 Tikhonov Regularization
5.1 Optimality Conditions for RKHS
5.2 RKHS Regularization Examples
3.3 Functions with Auxiliary Parameters
6 L1 Regularizations
7 Representer Theorem
3.4 Infimal Convolution, Indicator and Support Functions
8 Learning the Kernel
9 Biased Regularizations
10 Low–Rank Approximations of Kernel Matrices

Figure 1: A roadmap of this paper. The sections on convex analysis are in the left-hand column, while the "applications" are in the right-hand column.

If $S, S' \subset \mathbb{R}^n$ and $A \in \mathbb{R}^{m\times n}$, then $S+S' = \{y+y' : y \in S, y' \in S'\}$, $S-S' = \{y-y' : y \in S, y' \in S'\}$, $SS' = \{yy' : y \in S, y' \in S'\}$, and $AS = \{Ay : y \in S\}$. In particular, $A\mathbb{R}^n$ is the column space of A. We denote the topological interior of $S \subset \mathbb{R}^n$ by $\mathrm{int}(S)$. A cone is a set S with the property that $\mathbb{R}_{\geq 0}S \subset S$.

We write $\mathbb{B}_p \subset \mathbb{R}^n$ where $\mathbb{B}_p = \{y \in \mathbb{R}^n : ||y||_p \leq 1\}$ where $||\cdot||_p$ is the p-norm. We write $A^\dagger$ for the pseudoinverse of a matrix A.

3. Convex Analysis

In this section, we develop the necessary topics in convex analysis, including the needed elements of Fenchel duality theory. All results in this section can be found in Borwein and Lewis (2000) and

Rockafellar and Wets (2004), although in some cases we have substituted less general versions of the results that are sufficient for our purposes, and in some cases we have elaborated (with proofs) ideas that are introduced as exercises in these books.

3.1 Closed Convex Functions

Definition 1 *Given a function* $f:\mathbb{R}^n \to [-\infty,\infty]$, *the* epigraph *of* f, *epi* f, *is defined by*

$$epi\ f = \{(y,e) : e \geq f(y)\} \subset \mathbb{R}^n \times \mathbb{R}.$$

We say f *is* closed *or* convex *if epi* f *is closed, or convex.*
We define dom $f = \{y \in \mathbb{R}^n : f(y) < \infty\}$.
We say f *is* proper *when dom* $f \neq \emptyset$ *and* $f > -\infty$ *(i.e.,* $\forall y, f(y) > -\infty$*).*

(Some texts consider $f:\mathbb{R}^n \to (-\infty,\infty]$, whereupon $f > -\infty$ is automatic.)

We will mostly be considering f which do not take the value $-\infty$ (such functions are somewhat pathological), in which case f being proper is equivalent to dom $f \neq \emptyset$. Allowing f to take the value of ∞ merely allows some portions of epi f to have no projection onto $\mathbb{R}^n$ (dom f is that projection). One way of viewing constrained minimization problems is as unconstrained problems with an objective function that can take the value ∞. Indicator functions (introduced in Section 3.4) are a device for this purpose. We will not do arithmetic involving ∞ except where the result is unambiguous (e.g., $\infty + 1 = \infty$).

The functions of primary interest to us are closed, convex, proper functions. We will call such a function a *ccp* function.

Definition 2 *Given* $f:\mathbb{R}^n \to (-\infty,\infty]$, *we define the set* $\operatorname{argmin}_{y\in\mathbb{R}^n} f(y)$ *as follows,*

$$\operatorname*{argmin}_{y\in\mathbb{R}^n} f(y) = \begin{cases} \mathbb{R}^n & \inf_{y\in\mathbb{R}^n} f(y) = \infty \\ \{y : f(y) = f_0\} & \inf_{y\in\mathbb{R}^n} f(y) = f_0 \in \mathbb{R} \\ \emptyset & \inf_{y\in\mathbb{R}^n} f(y) = -\infty \end{cases}$$

with symmetrical definitions for argmax *when needed.*

If f is proper, then the first case cannot occur. The occurrence of the middle case (given f proper) is equivalent to f being bounded from below. However, even in the second case, $\operatorname{argmin}_y f(y)$ may still be empty.

The notion of a supporting hyperplane gives us a non-smooth generalization of the gradient, called a *subgradient*.

Definition 3 (subgradients and subdifferentials) *If* $f:\mathbb{R}^n \to (-\infty,\infty]$ *is convex and* $y \in$ *dom* f, *then* $\phi \in \mathbb{R}^n$ *is a* subgradient *of* f *at* y *if it satisfies* $\phi^t z \leq f(y+z) - f(y)$ *for all* $z \in \mathbb{R}^n$.

The set of all such ϕ *is the* subdifferential *and denoted* $\partial f(y)$. *By convention,* $\partial f(y) = \emptyset$ *if* $y \notin$ *dom* f.

∂f is a function ($y \to \partial f(y)$) whose values are convex sets. If f is differentiable at y, then $\partial f(y)$ contains a single point, the gradient, but not so if non-differentiable at y. It is possible for the subdifferential of a convex function to be $\emptyset$ when $y \in$ dom f (for example, $f(y) = -\sqrt{y}$ has $\partial f(0) = \emptyset$ even though $f(0) = 0$). However, $\partial f(y) \neq \emptyset$ for $y \in \operatorname{int}(\text{dom } f)$ (by Theorem 3.1.8 of Borwein and Lewis, 2000).

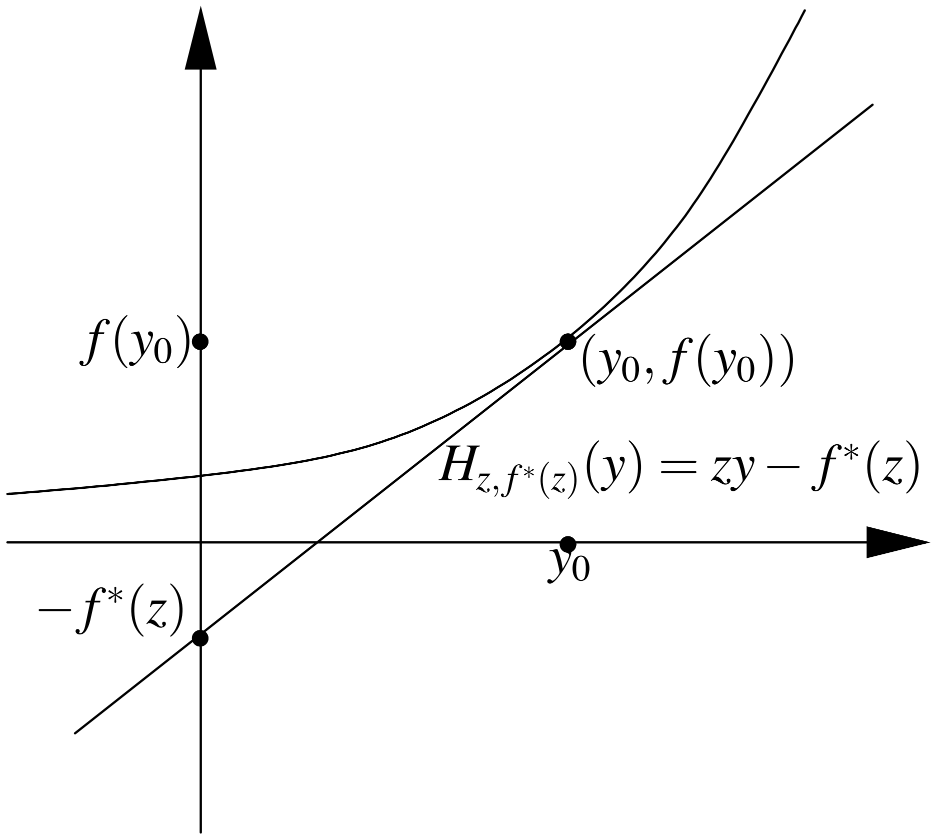


Figure 2: A graphical illustration of the Fenchel-Legendre conjugate.

3.2 Fenchel Duality

Central to Fenchel duality is the Fenchel-Legendre conjugate,

Definition 4 (Fenchel-Legendre conjugate) *Given a function* $f : \mathbb{R}^n \to [-\infty, \infty]$*, the* Fenchel-Legendre conjugate *is*

$$f^*(z) = \sup_y \{y^t z - f(y)\}. \tag{7}$$

Table 1 lists a few general, easily derived conjugation identities.

For a convex function f of one variable, one can get a sense of what the conjugate and subgradient look like by examining a graph of $f(y)$. For a given y, one finds a point $(y_0, f(y_0))$ (on the boundary of epi f) such that epi f is supported at y_0 by a line of the form $H_{z,c}(y) = zy - c$ (i.e., z is the slope of a tangent line of epi f at $(y_0, f(y_0))$). If such a supporting line exists, then $f^*(z) = c$, otherwise $f^*(z) = \infty$. Because f is convex, at most one such supporting line can exist; if f is linear in a neighborhood of y_0 then $H_{z,c}$ supports f at multiple points. Figure 2 illustrates the idea.

$f = g^*$	$g = f^*$	qualifiers
$f(y)$	$g(z)$	
$h(y)+c$	$h^*(z)-c$	
$h(y)-a^ty$	$h^*(z+a)$	
$ah(y)$	$ah^*(z/a)$	$a>0$
$h(A^{-1}y+b)$	$h^*(A^tz)-b\cdot z$	A non-singular
$\frac{1}{2}y^tAy$	$\frac{1}{2}z^tA^{-1}z$	A symmetric positive definite
v^ty+b	$\delta_{\{v\}}(z)-b$	

Table 1: Some conjugates and properties of conjugation.

More generally, we can work in terms of affine functions and epigraphs. Equation 7 is equivalent to

$$\text{epi } f^* = \bigcap_{y\in\text{dom } f} \text{epi } H_{y,f(y)}.$$

Being an arbitrary intersection of closed and convex sets, epi f^* is closed and convex. Thus, f^* is closed and convex even if f is neither. Additionally, for $z \in$ dom f^*, by (7), we have $H_{z,f^*(z)}(y) = z^ty - f^*(z) \le f(y)$ for all y, and hence,

$$\text{epi } f \subset \bigcap_{z\in\text{dom } f^*} \text{epi } H_{z,f^*(z)},$$

holding with equality when f is closed and convex (Theorem 4.2.1 of Borwein and Lewis, 2000).

Theorem 5 (biconjugation) $f : \mathbb{R}^n \to (-\infty,\infty]$ *is closed and convex iff* $f^{**} = f$.

In particular, conjugation is a bijection between ccp functions.

The supremum in (7) is attained if and only if $H_{z,f^*(z)}(y) = f(y)$ for some y. Thus, $z \in \partial f(y)$ is equivalent to $f(y) = y^tz - f^*(z)$ (Theorem 3.3.4 of Borwein and Lewis, 2000).

Theorem 6 (Fenchel-Young) *Let* $f : \mathbb{R}^n \to (-\infty,\infty]$ *be convex.* $\forall y,z \in \mathbb{R}^n$,

$$f(y) + f^*(z) \ge y^tz$$

with equality holding iff $z \in \partial f(y)$.

Combining the previous results, if f is closed and convex then $z \in \partial f(y) \Leftrightarrow y \in \partial f^*(z)$, and if $z \in \text{int}(\text{dom } f^*)$ the supremum in (7) is attained.

Fenchel duality can be motivated from Theorem 6 by considering two functions simultaneously. Given convex $f,g : \mathbb{R}^n \to (-\infty,\infty]$

$$f(y) + f^*(z) - y^tz \ge 0 \tag{8}$$
$$g(y) + g^*(-z) - y^t(-z) \ge 0. \tag{9}$$

Summing the above inequalities and minimizing,

$$f(y) + g(y) + f^*(z) + g^*(-z) \ge 0 \tag{10}$$
$$\inf_y\{f(y) + g(y)\} + \inf_z\{f^*(z) + g^*(-z)\} \ge 0. \tag{11}$$

If $\exists y,z \in \mathbb{R}^n$ such that (10) is an equality, then clearly (11) holds with equality as well and the individual infima are attained. Moreover, (8) and (9) become equalities and thus $z \in \partial f(y)$ and $-z \in \partial g(y)$ (and $y \in \partial f^*(z)$ and $y \in \partial g^*(-z)$, if f,g are ccp). We now quote the fundamental theorem of Fenchel duality, which supplies sufficient conditions for (11) to hold and for either of the infima to attain. Our statement is less general than Theorem 3.3.5 of Borwein and Lewis (2000), but will serve for our purposes.

Theorem 7 (Fenchel duality) *Let $f,g : \mathbb{R}^n \to (-\infty,\infty]$ be convex with $f+g$ bounded below. If $0 \in int(dom\ f - dom\ g)$, then (11) is an equality and the infimum of $\inf_z\{f^*(z)+g^*(-z)\}$ is attained.*

The topological sufficiency condition, $0 \in \text{int}(\text{dom } f - \text{dom } g)$, is stronger than $\text{dom } f \cap \text{dom } g \neq \emptyset$ (which is necessary for $f+g$ to be proper) and weaker than $\text{dom } f \cap \text{int}(\text{dom } g) \neq \emptyset$ or $\text{int}(\text{dom } f) \cap \text{dom } g \neq \emptyset$ (which is, in practice, easier to check).

Corollary 8 *Let $f,g : \mathbb{R}^n \to (-\infty,\infty]$ be ccp with $f+g$ bounded below. If $0 \in int(dom\ f^* + dom\ g^*)$, then (11) is an equality and the infimum of $\inf_y\{f(y)+g(y)\}$ is attained.*

Proof We apply Theorem 7 to $f^*(z)$ and $g^*(-z)$, noting that $\text{dom } g^*(-z) = -\text{dom } g^*(z)$, and that $f+g$ bounded below implies $f^*(z)+g^*(-z)$ is bounded below, by Equation 11. This shows that

$$\inf_z\{f^*(z)+g^*(-z)\} + \inf_y\{f^{**}(y)+g^{**}(y)\} \quad \geq \quad 0$$

is an equality; applying Theorem 5 proves the result. ■

Combining the above two corollaries yields the following variant which we will later apply to learning problems.

Corollary 9 *Let $f,g : \mathbb{R}^n \to (-\infty,\infty]$ be ccp with $f+g$ bounded below. If $0 \in int(dom\ f - dom\ g)$ or $0 \in int(dom\ f^* + dom\ g^*)$, then*

$$\inf_{y,z}\{f(y)+g(y)+f^*(z)+g^*(-z)\} = 0,$$

and all minimizers y,z satisfy the complementarity equations:

$$\begin{aligned} f(y) - y^t z + f^*(z) &= 0 \\ g(y) + y^t z + g^*(-z) &= 0. \end{aligned}$$

Additionally, if $0 \in int(dom\ f - dom\ g)$ and $0 \in int(dom\ f^* + dom\ g^*)$ then a minimizer (y,z) exists.

We are primarily interested in examples where f and g are ccp, both bounded below, and satisfy both sufficiency conditions. In this case, we see that the minimality conditions of (11) are given by a pair of coupled complementarity equations, each being dependent on only one of the two functions f and g. In the simple case where f and g are both differentiable, these complementary equations are nothing more than $z = \nabla f(y)$ and $-z = \nabla g(y)$, which is clearly the minimality condition for $f(y)+g(y)$. The value of these relations is their generality to non-smooth functions. In our applications to learning, we will be considering the above equations with the regularizer and the loss function taking on the roles of f and g.

3.3 Functions with Auxiliary Parameters

We are frequently interested in functions of the form $h'(y) = \inf_u h(y,u)$. If the infimum is attained for all y where $h'(y)$ is finite, then we say h' is *exact*. We will study the properties of h' through those of h.

Lemma 10 *If $h : \mathbb{R}^n \times \mathbb{R}^m \to [-\infty, \infty]$ with $h'(y) = \inf_u h(y,u)$ then $h'^*(z) = h^*(z,0)$.*

Proof $h^*(z,0) = \sup_{y,u}\{y^t z - h(y,u)\} = \sup_y\{y^t z - h'(y)\} = h'^*(z)$. ■
It is important to note that $h(y,u)$ being convex in y for fixed u does *not* guarantee that h' is convex. If h is ccp then h' is convex and dom $h' \neq \emptyset$, however, this does not guarantee that h' is exact, closed, or that $h' > -\infty$ (i.e., that h' is proper). We can obtain such guarantees by studying projections of dom h^* onto a subset of its variables.

Lemma 11 *Let $h : \mathbb{R}^n \times \mathbb{R}^m \to (-\infty, \infty]$ be ccp with $h'(y) = \inf_u h(y,u)$. If $W = \{w \in \mathbb{R}^m : \exists z \in \mathbb{R}^n, (z,w) \in dom\ h^*\}$ then $0 \in W \Rightarrow \forall y, h'(y) > -\infty$.*

Proof $\exists y, h'(y) = -\infty \Rightarrow \forall z, h^*(z,0) = \infty \Rightarrow 0 \notin W$. ■

Corollary 12 *Let h, h' be as in Lemma 11.*

$$z \in \partial h'(y) \textit{ and } h'(y) = h(y,u) \Leftrightarrow (z,0) \in \partial h(y,u).$$

Proof If $h'(y) - y^t z + h'^*(z) = 0$ and $h'(y) = h(y,u)$ then $h(y,u) - y^t z + h^*(z,0) = 0$ and thus $(z,0) \in \partial h(y,u)$. Conversely, if $h(y,u) - y^t z + h^*(z,0) = 0$, since $h(y,u) \geq h'(y)$, we have $0 \geq h'(y) - y^t z + h'^*(z)$. Thus $h'(y) - y^t z + h'^*(z) = 0$ and $h'(y) = h(y,u)$. ■

Lemma 13 *Let h, h' be as in Lemma 11. If $0 \in int(W)$ then h' is ccp and exact.*

Proof For fixed $y \in$ dom h', define $g_y(y', u) = \begin{cases} 0 & y' = y \\ \infty & \text{else} \end{cases}$. It is straightforward to see that $g_y^*(z,w) = \begin{cases} y^t z & w = 0 \\ \infty & \text{else} \end{cases}$, and dom $g_y^* = \mathbb{R}^n \times \{0\}^m$. Hence, dom h^* + dom $g_y^* = \mathbb{R}^n \times W$, and Corollary 8 applies:

$$\begin{aligned} \inf_u h(y,u) &= \inf_{d,u}\{h(d,u) + g_y(d,u)\} \\ &= -\inf_{z,w}\{h^*(z,w) + g_y^*(-z,-w)\} \\ &= -\inf_z\{h^*(z,0) - y^t z\} \\ &= \sup_z\{y^t z - h^*(z,0)\} \\ &= h'^{**}(y), \end{aligned}$$

and there exists d,u which attain $\inf_{d,u}\{h(d,u) + g(d,u)\}$, hence $h(y,u) = h'(y) = h'^{**}(y)$. ■

3.4 Infimal Convolution, Indicator and Support Functions

We introduce the notion of *infimal convolution*, an idea which will play a key role throughout this work.

Definition 14 (infimal convolution) *For $f,g:\mathbb{R}^n \to (-\infty,\infty]$, we define $f\star g:\mathbb{R}^n \to [-\infty,\infty]$, the* infimal convolution *of f and g, by*

$$(f\star g)(y) = \inf_{y'}\{f(y-y')+g(y')\}. \tag{12}$$

We say $f\star g$ is *exact* if the infimum of (12) is attained whenever $(f\star g)(y)$ is finite. If $f\star g$ is exact and $(f\star g)(y)$ is finite, we write $(f\star g)(y) = f(y-y')+g(y')$, implicitly defining y' as a minimizer of (12). The following theorem relates optimality conditions for f and g to optimality conditions for $f\star g$.

Theorem 15 *Let $f,g:\mathbb{R}^n \to (-\infty,\infty]$ be ccp.*

- *$(f\star g)^*(z) = f^*(z)+g^*(z)$. If $0\in$ dom $f^* -$ dom g^*, then $f\star g > -\infty$.*
- *$z\in\partial(f\star g)(y)$ and $(f\star g)(y) = f(y-y')+g(y') \Leftrightarrow z\in\partial f(y-y')\cap\partial g(y')$.*
- *If $0\in$ int(dom $f^* -$ dom g^*), then $f\star g = (f^*+g^*)^*$ and is exact (as well as ccp).*

Proof Let $h(y,y') = f(y-y')+g(y')$. $(f\star g)(y) = \inf_{y'} h(y,y')$, hence $f\star g$ is convex. It is straightforward to show that $h^*(z,z') = f^*(z)+g^*(z+z')$. Lastly, define $W = \{z'\in\mathbb{R}^n : (z,z')\in \text{dom } h^*\} = \text{dom } f^* - \text{dom } g^*$. With these results in place, we specialize the previous results.

The first claim is by Lemma 11 and the third by Lemma 13. The second is Corollary 12 with the additional observation:

$$\begin{aligned}
h(y,y') - y^t z + h^*(z,0) &= 0\\
\Leftrightarrow f(y-y') + g(y') - z^t(y-y'+y') + f^*(z) + g^*(z) &= 0\\
\Leftrightarrow \left[f(y-y') - z^t(y-y') + f^*(z)\right] + \left[g(y') - z^t(y') + g^*(z)\right] &= 0
\end{aligned}$$

Since the two bracketed terms are non-negative (by Theorem 6), the last line is equivalent to $f(y-y') - z^t(y-y') + f^*(z) = g(y') - z^t(y') + g^*(z) = 0$. ■

Many functions of interest can be expressed in terms of infimal convolutions of simpler functions.

There are a number of useful auxiliary functions and sets one may define relative to a given set C.

Definition 16 (indicator functions, support functions, and polarity) *For any non-empty set $C\subset\mathbb{R}^n$, the* indicator function *δ_C, the support function σ_C, and the* polar *of C, C° are given by*

$$\begin{aligned}
\delta_C(y) &= \begin{cases} 0 & y\in C\\ \infty & y\notin C\end{cases}\\
\sigma_C(y) &= \sup_{z\in C} z^t y\\
C^\circ &= \{z\in\mathbb{R}^n : \forall y\in C, y^t z \le 1\}.
\end{aligned}$$

Indicator functions allow us to work entirely with unconstrained functions—a problem of the form "minimize $f(y)$ subject to $y \in C$" becomes "minimize $f(y) + \delta_C(y)$." The support function $\sigma_C(y)$ has a simple interpretation as the largest projection of any element of C onto the line generated by y. Indicator functions, support functions, and polars are closely related, as the following lemma shows.

Lemma 17 *Let $C \subset \mathbb{R}^n$ be non-empty. σ_C is ccp, $\delta^*_C = \sigma_C$, C° is closed and convex, and $0 \in C^\circ$.*

Proof C non-empty implies $\sigma_C(0) = 0$, so σ_C is proper. Since epi $\sigma_C = \bigcap_{y \in C}$ epi $H_{y,0}$ and arbitrary intersections of closed and convex sets are closed and convex, σ_C is ccp.

By definition, $\delta^*_C(z) = \sup_y \{y^t z - \delta_C(y)\} = \sup_{y \in C} y^t z = \sigma_C(z)$.

$C^\circ = \{z \in \mathbb{R}^n : \sigma_C(z) \le 1\}$ is closed and convex, since it is the level set of a closed, convex function, and $\sigma_C(0) = 0$ implies $0 \in C^\circ$. ■

If C is closed, convex and non-empty, then δ_C is ccp, and Theorem 6 applies:

$$\begin{aligned} z \in \partial\delta_C(y) \Leftrightarrow y \in \partial\sigma_C(z) \quad &\Leftrightarrow \quad \delta_C(y) - y^t z + \sigma_C(z) = 0 \\ &\Leftrightarrow \quad y \in C, \forall y' \in C, z^t(y' - y) \le 0. \end{aligned}$$

If C is a cone then $\sigma_C = \delta_{C^\circ}$, and $C^\circ = \{z \in \mathbb{R}^n : \forall y \in C, z^t y \le 0\}$. If C is a vector subspace (a special case of a cone), then $C^\circ = C^\perp = \{z \in \mathbb{R}^n : \forall y \in C, z^t y = 0\}$.

Lemma 18 *Let $A, B \subset \mathbb{R}^n$ be closed, convex and non-empty.*

$$\begin{aligned} \delta_{A+B} &= \delta_A \star \delta_B & \sigma_{A+B} &= \sigma_A + \sigma_B \\ \delta_{A\cap B} &= \delta_A + \delta_B & \sigma_{A\cap B} &= \sigma_A \star \sigma_B \quad (\textit{if } 0 \in \textit{int}(A - B)) \\ \sigma_{A\cup B} &= \sigma_{A\oplus B} & \delta^{**}_{A\cup B} &= \delta_{A\oplus B} \end{aligned}$$

where $A \oplus B$ is the closure of the convex hull of $A \cup B$.

Proof The identities for $\delta_{A\cap B}$, δ_{A+B}, and $\sigma_{A\cup B}$ are easily shown. The others are from conjugation, (with $\sigma_{A\cap B}$ requiring Theorem 15, and hence the sufficiency condition). ■

A number of functions can be built up from support functions. For example, any norm can be defined as the support function of a closed convex set (namely, the unit ball of the associated dual norm). Table 2 contains common examples.

3.5 Summary

Figure 3 provides a concise summary of some of the most important ideas we have presented in this section. In the summary, we ignore necessary technical conditions, but we emphasize that all the applications of convex analysis in this paper refer to and demonstrate the relevant technical conditions. We believe this summary will be useful, especially to readers unfamiliar with the abstract theory of convex functions and conjugacy.

Keep in mind that if f and g are differentiable, much of the theory given here reduces to finding a y such that $\nabla f(y) + \nabla g(y) = 0$. In a very real sense, the entire purpose of the development in this section is to extend intuitions about the unconstrained optimization of differentiable functions to the constrained and non-differentiable setting. This is crucial, as the only unconstrained differentiable examples we are aware of in learning theory are regularized least squares and logistic regression.

C	$\sigma_C(y)$
$\lambda C, \lambda \geq 0$	$\lambda \sigma_C(y)$
AC	$\sigma_C(A^t y)$
$\mathbb{B}_p$	$\lVert y \rVert_{\frac{p}{p-1}}$
$\mathbb{R}^n$	$\delta_{\{0\}}(y)$
$\mathbb{R}^n_{\geq 0}$	$\delta_{\mathbb{R}^n_{\leq 0}}(y)$
$[-1,1]^n$	$\sum_i \lvert y_i \rvert$
$[0,1]^n$	$\sum_i (y_i)_+$
K	$\delta_{K^\circ}(y)$

Table 2: Support functions for common convex sets. C and C' are arbitrary non-empty closed convex sets. K is an arbitrary non-empty closed convex cone. A is an arbitrary matrix.

Summary of Key Convex Analysis Concepts

- **(Fenchel-Legendre Conjugate)** $f^*(z) = \sup_y \{y^t z - f(y)\}$.
- **(Biconjugation)** $f^{**} = f$.
- **(Fenchel-Young Theorem)** $f(y) - y^t z + f^*(z) \geq 0$, and $z \in \partial f(y) \Leftrightarrow y \in \partial f^*(z) \Leftrightarrow f(y) - y^t z + f^*(z) = 0$.
- **(Fenchel Duality)** The minimizer of $f(y) + g(y)$ satisfies

$$\begin{aligned} f(y) - y^t z + f^*(z) &= 0 \\ g(y) + y^t z + g^*(-z) &= 0 \end{aligned}$$

  for some z which also minimizes $f^*(z) + g^*(-z)$.
- **(Auxiliary Parameters:** $h'(y) = \inf_u h(y,u)$**)** $h'^*(z) = h^*(z,0)$.
- **(Infimal Convolutions:** $(f \star g)(y) = \inf_{y'} \{f(y-y') + g(y')\}$**)** $(f \star g)^* = f^* + g^*$.
- **(Indicator and Support Functions)** $\delta_C(y) = \begin{cases} 0 & y \in C \\ \infty & y \notin C \end{cases}$, and $\sigma_C(z) = \sup_{y \in C} y^t z$. $\delta_C^* = \sigma_C$. If C is a vector space, $\delta_C^* = \delta_{C^\perp}$.

Figure 3: Summary of key notions of convex conjugacy. In this figure, we assume that all necessary technical conditions (convexity and closedness of functions, exactness of infimal convolutions, etc.) are met; the technical conditions are given in full in the text. All examples given in this paper satisfy the necessary technical conditions. Of course, when considering new examples, the conditions must be checked.

4. Tikhonov Regularization

We now specialize the general theory of Fenchel duality to the inductive learning scenario.

Definition 19 (loss functions and regularization) *A* loss function *is any closed convex function* $v : \mathbb{R} \to (\infty, \infty]$*, which is bounded below and finite at* 0*.*

A regularization *is any closed convex function* $R : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$*, such that* $R(0) = 0$*.*

Typically, a loss function, v_i, represents the penalty for a value mismatching some prescribed value or set of values (e.g., $v_i(y_i) = (y_i - Y_i)^2$), while a regularization represents a measure of *non-smoothness* among a set of values $y_1, \ldots, y_n$.

Clearly, regularizations and loss functions are closed under addition. We can also show that the conjugates of regularizations and loss functions are, respectively, regularizations and loss functions.

Lemma 20 *If* v *is a loss function, then* v^* *is a loss function. If* R *is a regularization, then* R^* *is a regularization.*

Proof Since v^* and R^* are closed and convex, we need only show that v^* is bounded below and finite at 0 and that $R^*(z) \geq 0$ with $R^*(0) = 0$.

v is bounded below, so $v^*(0) = -\inf_y v(y)$ is finite. $v(0) = -\inf_z v^*(z)$ is finite, so v^* is bounded below.

Since $R(0) = -\inf_z R^*(z) = 0$, R^* is non-negative and $R^*(0) = -\inf_y R(y) = 0$. ■

As a consequence, regularizations and losses are closed under infimal convolution.

The following lemma shows that if the loss function can be decomposed over data points, then its conjugate can likewise be decomposed.

Lemma 21 *Let* $V : \mathbb{R}^n \to (-\infty, \infty]$ *be given by* $V(y) = \sum_{i=1}^n v_i(y_i)$ *for loss functions* v_i*. Then* $V^*(z) = \sum_{i=1}^n v_i^*(z_i)$*.*

Proof A direct consequence of the definition,

$$\begin{aligned} V^*(z) &= \sup_y \left\{ y^t z - \sum_i v_i(y_i) \right\} \\ &= \sum_i \sup_{y_i} \{ y_i z_i - v_i(y_i) \} \\ &= \sum_i v_i^*(z_i). \end{aligned}$$

■

We are now able to state the main theorem that we will use to study regularization problems.

Theorem 22 (regression Fenchel duality) *Let* $R : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ *be a regularization and* $V(y) = \sum_{i=1}^n v_i(y_i)$ *for loss functions* $v_i : \mathbb{R} \to \mathbb{R}$*. If* $0 \in int(dom\ R - dom\ V)$ *or* $0 \in int(dom\ R^* + dom\ V^*)$ *then*

$$\inf_{y,z} \{ R(y) + V(y) + R^*(z) + V^*(-z) \} = 0$$

with all minimizers y, z *satisfying the complementarity equations:*

$$\begin{aligned} R(y) - y^t z + R^*(z) &= 0 && (13) \\ v_i(y_i) + y_i z_i + v_i^*(-z_i) &= 0. && (14) \end{aligned}$$

Additionally, if $0 \in int(dom\ R - dom\ V)$ *and* $0 \in int(dom\ R^* + dom\ V^*)$ *then a minimizer exists.*

loss	dual loss	optimality condition
$v(y)$	$v^*(-z)$	$v(y)+yz+v^*(-z)=0$
$f(Y-y)$	$f^*(z)-zY$	$f(Y-y)+(Y-y)(-z)+f^*(z)=0$
$f(1-yY)$	$f^*\left(\frac{z}{Y}\right)-\frac{z}{Y}$	$f(1-yY)+(1-yY)\frac{-z}{Y}+f^*\left(\frac{z}{Y}\right)=0$
$\frac{1}{2}y^2$	$\frac{1}{2}z^2$	$y+z=0$
$\|y\|$	$\delta_{[-1,1]}(z)$	$\|y\|+yz=0, z\in[-1,1]$
$(y)_+$	$\delta_{[-1,0]}(z)$	$(y)_+ + yz=0, z\in[-1,0]$
$\frac{1}{2}(y)_+^2$	$\delta_{\mathbb{R}_{\leq 0}}(z)+\frac{1}{2}z^2$	$(y)_+ + z=0, z\leq 0$
$(\|y\|-\varepsilon)_+$	$\delta_{[-1,1]}(z)+\varepsilon\|z\|$	$\left\{\begin{array}{ll} \|y\|\geq\varepsilon & z+\operatorname{sign}(y)=0 \\ \|y\|\leq\varepsilon & z\in[-1,1] \end{array}\right\}$
$\frac{1}{2}(Y-y)^2$	$\frac{1}{2}z^2-zY$	$y+z=Y$
$\|Y-y\|$	$\delta_{[-1,1]}(z)-zY$	$\|Y-y\|=(Y-y)z, z\in[-1,1]$
$\|1-yY\|$	$\delta_{[-1,1]}\left(\frac{z}{Y}\right)-\frac{z}{Y}$	$\|1-yY\|=(1-yY)\frac{z}{Y}, \frac{z}{Y}\in[-1,1]$
$(1-yY)_+$	$\delta_{[0,1]}\left(\frac{z}{Y}\right)-\frac{z}{Y}$	$(1-yY)_+=(1-yY)\frac{z}{Y}, \frac{z}{Y}\in[0,1]$
$\frac{1}{2}(1-yY)_+^2$	$\delta_{\mathbb{R}_{\geq 0}}\left(\frac{z}{Y}\right)+\frac{1}{2}\frac{z^2}{Y^2}-\frac{z}{Y}$	$(1-yY)_+=\frac{z}{Y}, \frac{z}{Y}\geq 0$
$(\|Y-y\|-\varepsilon)_+$	$\delta_{[-1,1]}(z)+\varepsilon\|z\|-zY$	$\left\{\begin{array}{ll} \|Y-y\|\geq\varepsilon & z=\operatorname{sign}(Y-y) \\ \|Y-y\|\leq\varepsilon & z\in[-1,1] \end{array}\right\}$
$\log(1+\exp(-yY))$	$\delta_{[0,1]}\left(\frac{z}{Y}\right)+\frac{z}{Y}\log\frac{z}{Y}+\left(1-\frac{z}{Y}\right)\log\left(1-\frac{z}{Y}\right)$	$1=(1+\exp(-yY))\frac{z}{Y}$

Table 3: A list of common loss functions and their local optimality conditions. Note that we are considering $v^*(-z)$, not $v^*(z)$. Also note that some of the loss functions can be further simplified under the assumption $Y\in\{-1,1\}$.

We identify the y_i as values taken by the learned function, scored by v_i, and the function values are penalized by $R(y)$. Equation 13 is a complementarity equation in $2n$ variables. Equation 14 is n independent complementarity equations in 2 variables. If R and v_i are differentiable these equations are equivalent to $z=\nabla R(y)$ and $-z_i=\frac{d}{dy}v_i(y_i)$.

4.1 Loss Functions

Table 3 recapitulates many of the loss functions seen in practice. In this paper, we deal exclusively with pointwise losses, so Table 3 is in terms of a single regression value y and a single training value Y with subscripts omitted.

The derivations in Table 3 can all be obtained without much effort from identities and definitions previously stated and earlier derivations in the table. It is helpful to observe that many loss functions can be expressed in terms of infimal convolutions of simple functions. For example, $(y)_+=\sigma_{[0,1]}(y)=(|\cdot|\star\delta_{\mathbb{R}_{\leq 0}})(y)$ and $(|y|-\varepsilon)_+=(|\cdot|\star\delta_{[-\varepsilon,\varepsilon]})(y)$.

It should be noted that these derivations can be checked graphically, since the losses are functions of a single variable. For example, Figure 4 shows a graph of $(1-y)_+$, the well-known *hinge loss*, with supporting hyperplanes. From the figure, it is plain that the only supporting hyperplanes are those with slopes in the interval $[-1,0]$, thus dom $f^*=[-1,0]$, and that $f^*(y)=y$ when $y\in[-1,0]$.

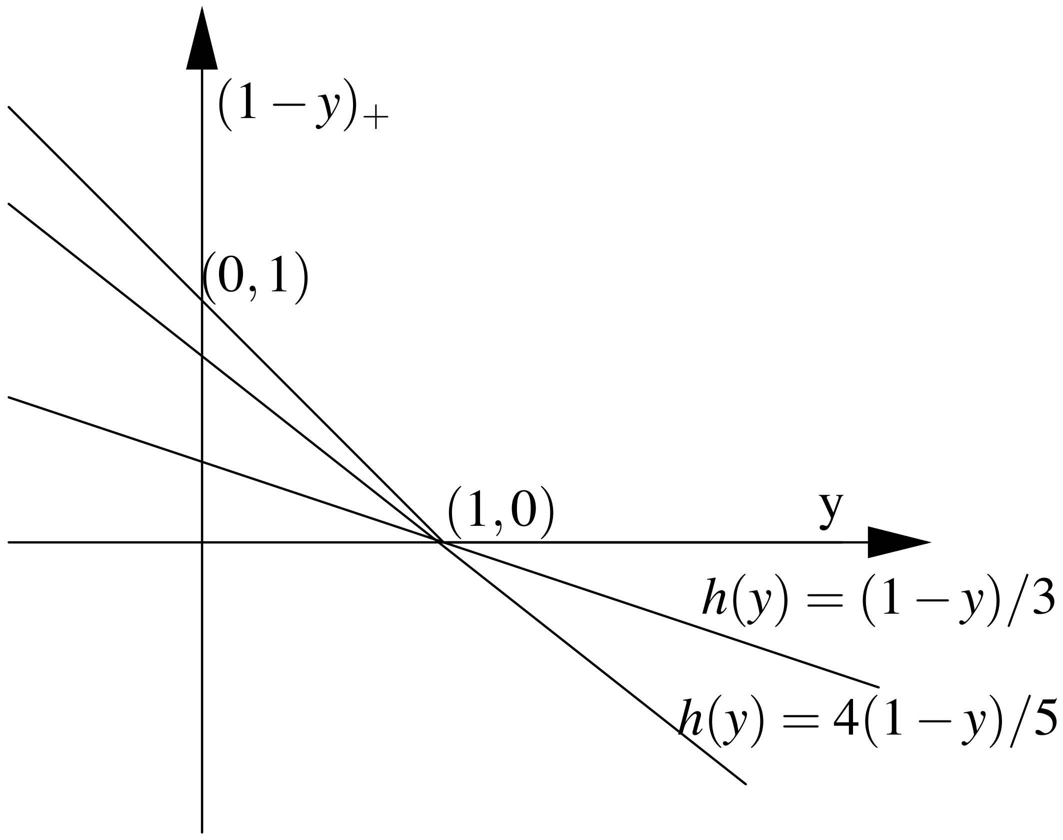


Figure 4: The graph of $(1-y)_+$ with a couple supporting hyperplanes.

4.2 Regularization Functions

By Lemma 20, if $R_1, R_2 : \mathbb{R}^n \rightarrow [0,\infty]$ are regularizations, then $R_1 + R_2$ and $R_1 \star R_2$ are, which gives two ways of building more complicated regularizations out of simpler ones. The basic building blocks of regularizations are indicator functions, support functions, and quadratic forms. The identities to keep in mind are $\sigma_C^* = \delta_C$, $Q_A^* = Q_{A^{-1}}$ for non-singular A, and $R_1 \star R_2 = (R_1^* + R_2^*)^*$ (with appropriate conditions from Theorem 15).

We might think of sums of regularizations as adding further penalties on the regression values, as $R \leq R + R'$. On the other hand, infimal convolutions add slack, $R \geq R \star R'$. Since these two operations are conjugates of each other, it is clear that any additional restriction in the primal results in extra freedom in the dual and vice versa.

5. Regularization in Reproducing Kernel Hilbert Spaces

We now turn to our primary example, Tikhonov regularization in a reproducing kernel Hilbert space (RKHS).

reg.	dual reg.	optimality condition
$R(y)$	$R^*(z)$	$R(y) - yz + R^*(-z) = 0$
$R_1 \star R_2(y)$	$R_1^*(z) + R_2^*(z)$	$R_1(y-y') - (y-y')^t z + R_1^*(z) = R_2(y') - y'^t z + R_2^*(z) = 0$
$R(A^t y)$	$R^*(A^\dagger z) + \delta_{A\mathbb{R}^n}(z)$	$R(A^t y) - y^t z + R(A^\dagger z) = 0, z \in A\mathbb{R}^n$
$\sigma_C(y)$	$\delta_C(z)$	$\sigma_C(z) = y^t z, z \in C$
$\sigma_C(A^t y)$	$\delta_{AC}(z)$	$\sigma_C(z) = y^t z, z \in C$
$Q_A(y) = \frac{1}{2} y^t A y$	$Q_{A^\dagger}(z) + \delta_{A\mathbb{R}^n}(z)$	$z = Ay$
$\|\|y\|\|_p$	$\delta_{\|\|z\|\|_q \leq 1}(z)$	$\|\|y\|\|_p = y^t z, \|\|z\|\|_q \leq 1, (\frac{1}{p} + \frac{1}{q} = 1)$
$\|\|A^t y\|\|_1$	$\delta_{A[-1,1]^n}(z)$	$\|\|A^t y\|\|_1 = y^t z, z \in A[-1,1]^n$
$\frac{1}{2}\|\|y\|\|_p^2$	$\frac{1}{2}\|\|z\|\|_q^2$	$\|\|y\|\|_p^2 = \|\|z\|\|_q^2 = y^t z, (\frac{1}{p} + \frac{1}{q} = 1)$

Table 4: Common regularization choices.

5.1 Optimality Conditions for RKHS Regularization

Recall that the "standard" approach in machine learning is to start with a Tikhonov regularization problem over an RKHS (2), to invoke the representer theorem to show that the solution can be expressed as a collection of coefficients (3), and to write optimization problems in terms of these coefficients. We start with a mathematical program in terms of the y_i, using a regularization $Q_{K^{-1}}$ (with K symmetric, positive definite); in this framework, we are able to state general optimality conditions and derive a very clean form of the representer theorem.

Specialized to the RKHS case, we are considering primal problems of the form

$$\inf_{y \in \mathbb{R}^n} \left\{ \frac{1}{2} \lambda y^t K^{-1} y + V(y) \right\},$$

where we have made the target labels Y_i implicit in V. The associated dual problem is

$$\inf_{z \in \mathbb{R}^n} \left\{ \frac{1}{2} \lambda^{-1} z^t K z + V^*(z) \right\}.$$

Equation 13 specializes to

$$\begin{aligned} \frac{1}{2}\lambda y^t K^{-1} y - y^t z + \frac{1}{2}\lambda^{-1} z^t K z &= 0 \\ \frac{1}{2}(y - \lambda^{-1} K z)^t (\lambda K^{-1} y - z) &= 0. \end{aligned}$$

and thus we obtain the optimality conditions $z = \lambda K^{-1} y$ (or $y = \lambda^{-1} K z$). This demonstrates that whenever we are performing regularization in an RKHS, the optimal y values at the training points can be obtained by multiplying the optimal dual variables z by $\lambda^{-1} K$, *for any convex loss function*.

5.2 RKHS Regularization Examples

In RKHS regularization, we have a regularizer $R(y) = \lambda Q_{K^{-1}}(y)$, with conjugate $R^*(z) = \lambda^{-1} Q_K(z)$, and optimality conditions $y = \lambda^{-1} K z$. By incorporating a loss function, we can derive well-known kernel machines.

5.2.1 REGULARIZED LEAST SQUARES

The *square* loss is given by $v(y_i) = \frac{1}{2}(Y_i - y_i)^2$. Although it is listed in Table 3, we derive the dual loss here, for pedagogical purposes:

$$v^*(-z) = \sup_{y'}\{-y'z - \frac{1}{2}(y' - Y)^2\}.$$

We see that $v^*(-z)$ is quadratic in y', and we can find the sup by taking the derivative:

$$\frac{\partial v^*}{\partial y'} = -z + (Y - y').$$

Setting the derivative to 0, we obtain $y = Y - z$, and substituting, we see that

$$v^*(-z) = -zY + \frac{1}{2}z^2.$$

The optimality equation $v(y_i) + y_i z_i + v^*(z_i)$ reduces to $y + z = Y$, so (13) and (14) become

$$\begin{aligned} y_i + z_i &= Y_i \\ \lambda^{-1}Kz &= y. \end{aligned}$$

Combined,

$$y + \lambda K^{-1}y = \lambda^{-1}Kz + z = Y,$$

leading to the standard regularized least squares formulation.

5.2.2 UNBIASED SVM

The support vector machine arises from the combination of RKHS regularization and the hinge loss $v(y) = (1 - yY)_+$. The dual of the hinge loss, $v^*(-z) = \delta_{[0,1]}(\frac{z}{Y}) - \frac{z}{Y}$, is most easily derived using a graphical approach (see Figure 4), but it can also be derived directly. Regularizing in an RKHS, we have the primal and dual problems:

$$\inf_{y \in \mathbb{R}^n} \left\{ \frac{1}{2}\lambda y^t K^{-1} y + \sum_i (1 - y_i Y_i)_+ \right\}$$

$$\inf_{z \in \mathbb{R}^n} \left\{ \frac{1}{2}\lambda^{-1} z^t K z + \sum_i \left(\delta_{[0,1]}\left(\frac{z_i}{Y_i}\right) - \frac{z_i}{Y_i} \right) \right\}.$$

We see how the δ function in the dual loss provides the well-known "box constraints" in the SVM dual.

Given $v(y), v^*(-z)$, it is also straightforward to derive the optimality condition

$$(1 - yY)_+ = (1 - yY)\frac{z}{Y} \quad \text{and} \quad \frac{z}{Y} \in [0, 1].$$

A case analysis on the sign of $(1-yY)$ yields

$$\begin{aligned}(1-y_iY_i)<0 &\iff \frac{z_i}{Y_i}=0\\(1-y_iY_i)=0 &\iff \frac{z_i}{Y_i}\in[0,1]\\(1-y_iY_i)>0 &\iff \frac{z_i}{Y_i}=1.\end{aligned}$$

With the regularization condition $y=\lambda^{-1}Kz$, we have derived the KKT conditions for the SVM. To obtain a more "standard" formulation, we could introduce variables $\alpha_i\equiv\frac{z_i}{Y_i}$, $c=\lambda^{-1}z$, eliminating y from the system (in favor of z and α), and introducing a "regularization constant" $C=\frac{1}{2\lambda}$.

In this section, we have derived an "unbiased" SVM. In practice, the SVM usually includes an unregularized bias term b. We show how to handle the bias term b in Section 9.1.

6. L1 Regularizations

Tikhonov regularization in a reproducing kernel Hilbert space is the most common form of regularization in practice, and has a number of mathematical properties which make it especially nice. In particular, the representer theorem makes RKHS regularization essentially the only case where we can start with a problem of optimizing an infinite-dimensional function in a function space, and obtain a finite-dimensional representation.[5] Nevertheless, other regularizations may be of interest. In these cases, we are generally forced to assume *a priori* that the functions we are looking for are coefficients in some finite dimensional space.

6.1 1-norm Support Vector Machines

As a consequence of the use of the hinge loss, support vector machines yield a sparse solution—points which live outside the margin have $c=z=0$. Nevertheless, because all points which are errors or live inside the margin are support vectors, in noisy problems, one generally finds that at least a constant fraction of the examples (lower-bounded by the Bayes error rate) are support vectors. Instead of using RKHS regularization, we can a priori choose a finite-dimensional function space of expansion coefficients of kernel functions around the training points $y=\sum_i K(x,x_i)c_i$, and regularize $||c||_1$. If we combine this idea with the hinge loss, we obtain the 1-norm support vector machine (Zhu et al., 2003).

We consider $R(y)=||K^{-1}y||_1$. If we use the hinge loss (or any other piecewise linear loss function), the resulting mathematical optimization problem is a linear program. The dual regularizer is $R^*(z)=\delta_{K^{-1}[-1,1]^n}(z)$. As usual, we can combine primal and dual regularizers with primal and dual loss functions. Note that in these formulations, the formula $c=\lambda^{-1}z$ *does not hold*: if we solved a problem in y and z, we would need to then solve $c=K^{-1}y$ to obtain c. In practice, we expect that algorithms based directly on c are probably the most useful.

5. Kernel-based algorithms are made practical by two factors: the representer theorem for RKHS's and kernels which make kernel products (and therefore the representations) easy to compute. One can construct RKHS's, for example, via embeddings, where learning is not practical because of a computationally difficult feature map and inner product. We thank a reviewer for pointing out this distinction.

6.2 Sparse Approximation and Support Vector Regression

In Girosi (1998), an "equivalence" between a certain sparse approximation problem and a modified form of SVM regression is developed. In slightly simplified form, the equivalence is quite easy to derive.

We begin by considering the sparse approximation problem. We are given *noiseless* samples $\{(X_1,Y_1),\ldots,(X_n,Y_n)\}$ from a target function f^t which is assumed to live in an RKHS $\mathcal{H}$. We will find a function which is a coefficient expansion around the data points: $f(x) = \sum_i K(x,x_i), c_i$. The goal is to minimize the distance between f^t and $f(x)$ in the RKHS norm,[6] while maintaining sparsity of the coefficient representation c:

$$\inf_{c\in\mathbb{R}^n}\left\{\frac{1}{2}||f^t-f||^2_{\mathcal{H}}+\varepsilon||c||_1\right\} = \frac{1}{2}||f^t||^2_{\mathcal{H}}+\inf_{c\in\mathbb{R}^n}\left\{\frac{1}{2}c^tKc-Y^tc+\varepsilon||c||_1\right\}.$$

where we used basic properties of RKHS and the assumption that $f^t(x)\in\mathcal{H}$ to expand $||f^t-f||^2_{\mathcal{H}}$.

Now we turn to a variant of support vector machine regression. A standard support vector regression machine is obtained when we use a loss $v(y_i) = (|Y_i-y_i|-\varepsilon)_+$, linearly penalizing points which lie outside a tube. Girosi instead considers a "hard margin" variant, where points are *required* to lie inside the tube. His primal problem is:

$$\inf_{y\in\mathbb{R}^n}\left\{\lambda Q_{K^{-1}}(y)+\delta_{[-\varepsilon,\varepsilon]^n}(Y-y)\right\}.$$

The dual to the "hard loss" $v(y_i) = \delta_{[-\varepsilon,\varepsilon]}(Y_i-y_i)$ is given by

$$\begin{aligned} v^*(-z_i) &= \sup_{y_i'}\{-y_i'z_i-\delta_{[Y_i-\varepsilon,Y_i+\varepsilon]}(y_i')\} \\ &= \max\{-(Y_i-\varepsilon)z_i,-(Y_i+\varepsilon)z_i\} \\ &= -Y_iz_i+\varepsilon|z_i|. \end{aligned}$$

Therefore, the dual problem is

$$\inf_{z\in\mathbb{R}^n}\left\{\lambda^{-1}Q_K(z)-Y^tz+\varepsilon||z||_1\right\} = \lambda\cdot\inf_{z\in\mathbb{R}^n}\left\{Q_K(\lambda^{-1}z)-Y^t(\lambda^{-1}z)+\varepsilon||\lambda^{-1}z||_1\right\}.$$

We see that the sparse approximation variant and the support vector regression variant have optimal values that differ by a constant $\frac{1}{2}||f^t||^2_{\mathcal{H}}$ and a positive multiplier λ, with $c=\lambda^{-1}z$. The sparsity control parameter ε becomes the width of the allowed tube in the regression problem. This makes sense: as we increase ε, we get more sparsity, but our predicted y values must become less tightly constrained in order to achieve that sparsity.[7]

6. In earlier treatments of sparse decomposition (such as Chen et al., 1995) the goal was to minimize functional distance in L_2 rather than $\mathcal{H}$; this distance (of course) needed to be approximated empirically.

7. Girosi considered biased regularizations with a free constant b. In order to incorporate this, he was forced to make additional assumptions on f^t and K. While these assumptions gave a formal equivalence between the sparse problem and the biased SVM regression quadratic program, we feel that the simplified version presented here illuminates the essential equivalence much more clearly.

7. Representer Theorem

We can generalize beyond the training points to talk about the predicted values at future points—our own version of the representer theorem. We consider a scenario where we have n training points and m additional unlabelled points and show how the y values at the unlabelled points can be determined via kernel expansions around the training points.

We begin by relating arbitrary regularizations in higher and lower-dimensional spaces.

Lemma 23 *Let $R : \mathbb{R}^{n+m} \to (-\infty, \infty]$ be a regularization, and let $R' = \inf_{y_2 \in \mathbb{R}^m} R(y_1, y_2)$. If $0 \in$ int(dom R^*), then R' is ccp and exact, R' is a regularization, and $R'^*(z_1) = R^*(z_1, 0)$. Additionally,*

$$y_1 \in \partial R'^*(z_1) \text{ and } R'(y_1) = R(y_1, y_2) \Leftrightarrow (y_1, y_2) \in \partial R^*(z_1, 0).$$

Proof By Lemmas 11 and 13, we have R' being ccp and exact. Clearly, $\inf_{y_1} R'(y_1) = R'(0) = 0$. The last claim comes from Corollary 12 and Theorem 6. ■

We specialize the lemma to the quadratic form regularizers associated with RKHS:

Corollary 24 *Let $K = \begin{pmatrix} K_{NN} & K_{NM} \\ K_{MN} & K_{MM} \end{pmatrix} \in \mathbb{R}^{(n+m)\times(n+m)}$ be symmetric positive definite where $K_{NN} \in \mathbb{R}^{n\times n}$. For all $y_1 \in \mathbb{R}^n$,*

$$\inf_{y_2 \in \mathbb{R}^m} Q_{K^{-1}}(y_1, y_2) = Q_{K_{NN}^{-1}}(y_1), \tag{15}$$

where the infimum is (uniquely) attained at $y_2 = K_{MN} K_{NN}^{-1} y_1$.

Proof Let $R(y_1, y_2) = Q_{K^{-1}}(y_1, y_2)$, and $R'(y_1) = \inf_{y_2} R(y_1, y_2)$. $R^*(z_1, z_2) = Q_K(z_1, z_2)$, thus dom $R^* = \mathbb{R}^{n+m}$ and R' is therefore closed and exact. $R'^*(z_1, 0) = Q_K(z_1, 0) = Q_{K_{NN}}(z_1)$, so $R'(y_1) = Q_{K_{NN}^{-1}}(y_1)$, hence (15). Given y_1, let z_1 satisfy $K_{NN} z_1 = y_1$. Then, $y_1 \in \partial R'^*(z_1)$. Also, $\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \in \partial R^*(z_1, 0)$ iff $y_2 = K_{MN} z_1$. By the last part of Lemma 23, $y_2 \in \operatorname{argmin}_{y_2} Q_{K^{-1}}(y_1, y_2)$. Since z_1 is unique given y_1, y_2 is unique. ■

We are now able to prove an optimization-centered variant of the representer theorem.

Theorem 25 *Let $X_1, \ldots, X_n, \ldots, X_{n+m} \in \mathbb{R}^d$ with $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ an symmetric positive definite kernel function. Let $K \in \mathbb{R}^{(n+m)\times(n+m)}$ be defined by $K_{ij} = k(X_i, X_j)$ for $1 \le i, j \le n+m$ with $K_{NN} \in \mathbb{R}^{n\times n}$ the upper left submatrix of K.*

For any function $V : \mathbb{R}^n \to (-\infty, \infty]$,

$$\inf_{y \in \mathbb{R}^{n+m}} \{Q_{K^{-1}}(y) + V(y_1, \ldots, y_n)\} = \inf_{y' \in \mathbb{R}^n} \left\{ Q_{K_{NN}^{-1}}(y') + V(y') \right\},$$

where $y_i = y'_i$ for $1 \le i \le n$ and $y_i = \sum_{j=1}^n k(X_i, X_j) c_j$ with $c = K_{NN}^{-1} y'$.

Proof Since the loss term V is independent of the last m components of y,

$$\begin{aligned} \inf_{y \in \mathbb{R}^{n+m}} \{Q_{K^{-1}}(y) + V(y_1, \ldots, y_n)\} &= \inf_{y' \in \mathbb{R}^n} \left\{ \inf_{y'' \in \mathbb{R}^m} Q_{K^{-1}}(y', y'') + V(y') \right\} \\ &= \inf_{y' \in \mathbb{R}^n} \left\{ \inf_{y'' \in \mathbb{R}^m} Q_{K_{NN}^{-1}}(y') + V(y') \right\}, \end{aligned}$$

by Corollary 24. ■

Our variant of the representer theorem shows that any point y_i which appears in the regularization but not the loss term can be determined from an expansion of the kernel function about the training points. Thus, solving the n variable optimization problem simultaneously "solves" all $n+m$ optimization problems with any additional set of m lossless points. This was made possible by the "subsetting" property of quadratic form regularizers, developed in Corollary 24.

An alternate way of thinking about this version of the representer theorem is that for regularization in an RKHS, the inductive and transductive cases "match": if we are given n training points and m testing points in advance, we can either train the standard n point optimization problem and use the coefficients c to compute the y values at the test points, or we can directly solve a larger optimization problem in which the test points appear in the regularization but not in the loss.

In RKHS regularization, we define $c \equiv K^{-1}y = \lambda^{-1}z$; this is justified by the representer theorem. We see that not only can we compute the values at test points, but the coefficients c have a direct and simple relation to the dual variables z. As we will see in later sections, when we explore other regularizations and kernel approximations, when we leave the RKHS setting, the simple relation between c and z is severed.

It is worthwhile to put our result in the context of more "standard" forms of the representer theorem. In its most general form (see Schölkopf et al. (2001) or Belkin et al. (2004) for proofs of this statement with a slightly different notation and emphasis), the representer theorem states that the optimal solution to an optimization problem of the form:

$$\min_{f \in \mathcal{H}} \left\{ \lambda ||f||_K^2 + F(f(X_1), \ldots, f(X_n)) \right\}, \tag{16}$$

where F is an arbitrary functional that depends only on the values f takes at the X_i, will have a solution of the form

$$f(x) = \sum_{i=1}^{n} c_i K(x, X_i), \tag{17}$$

and that the infinite-dimensional problem of finding an optimal $f \in \mathcal{H}$ can be reduced to the finite-dimensional problem of finding the optimal c_i. Traditionally, F is taken to be a loss function over labelled training points, but this is not required—for example, if the training set contains a mix of labelled and unlabelled points, F may be the sum of a loss function over the labelled points and some additional regularizer over the unlabelled points (see Belkin et al. (2004); Rahimi et al. (2005), and also Section 7.1; note that in Theorem 25 we define V as arbitrary and do not require it to be a loss function). The proof essentially proceeds by defining f to be a sum of kernel expansions (Equation 17) around the training points and a "remainder" which is orthogonal to these kernel expansions, and then showing that the orthogonal remainder must be the zero function. In our development, it is clear that solving

$$\min_{y \in \mathbb{R}^n} \left\{ y^t K^{-1} y + V(y) \right\}, \tag{18}$$

and defining $y = Kc$ will find the optimal solution to (16) of the form (17). Any function in the RKHS can be represented as a possibly infinite expansion of the form (17); Theorem 25 shows that if we add any (finite) set of additional points to the representation, none of the predicted training values change, and the expansion coefficients at the new points vanish. A simple limiting argument allows us to extend to infinite sets of additional points, demonstrating that the solution to (16) can be found by solving (18).

7.1 Semi-supervised and Transductive Learning

In this section we discuss in detail the somewhat subtle relationships between semi-supervised, transductive and inductive learning, quadratic regularizers, and the representer theorem.

In standard usage, inductive and semi-supervised learning are viewed as problems of taking a training set (fully labelled for the inductive case, partially unlabelled for the semi-supervised case) and learning a *function* that classifies new examples. In contrast, transductive learning is often viewed as the problem of predicting the labels at unlabelled points, given the locations of those points *in advance*. It must be understood that this distinction is somewhat artificial, based on notions of what we consider a function. In particular, given any transductive algorithm $\mathcal{A}$, we obtain a semi-supervised algorithm for classifying new points by rerunning $\mathcal{A}$ with all points seen so far whenever a prediction is needed. From this bird's-eye view, all problems about "out of sample" extensions for transductive algorithms (see for example Bengio et al. (2003)) vanish. This is a perfectly legitimate semi-supervised algorithm, in that it provides a *function* for computing the values at new points; however, because evaluating the function requires solving an optimization problem, many people are uncomfortable with this algorithm. At the present time, it seems that it is common usage in the machine learning community to accept predictive functions which are weighted sums of expansions around a fixed basis as legitimate semi-supervised solutions, but to reject predictive functions which require solving optimization problems as being merely transductive algorithms. This viewpoint is in some sense arbitrary, but also reflects genuine computational concerns; for the remainder of this section, we refer to transductive and inductive algorithms as the terms are commonly used.

Given an arbitrary transductive algorithm, it is possible to turn it into an inductive algorithm by embedding the optimization in an RKHS. An excellent example is manifold regularization Belkin and Niyogi (2003). As originally formulated, manifold regularization is defined only for points on a graph, and is therefore a transductive algorithm. In Belkin et al. (2004), the authors construct an inductive algorithm by considering the optimization problem:

$$\min_{y\in\mathbb{R}^{m+n}} \left\{\lambda_1 y^t K^{-1} y + \lambda_2 y^t L y + V(y_N)\right\}, \tag{19}$$

where we have n labelled and m unlabelled points, y_N are the predicted values at (just) the labelled points, and L is the graph Laplacian of (all) the data. The term $y^t Ly$ is an additional smoothness penalty on the predicted values that respects the notion that we expect the data to live on or near a low-dimensional manifold. By the standard representer theorem, solving the above optimization is equivalent to finding a function with minimal sum of its RKHS norm and the "loss function" $\lambda_2 y^t Ly + V(y_N)$. Similar extensions of straightforward RKHS regularizers with additional penalties are known for time series (Rahimi et al., 2005) and structured prediction (Altun et al., 2005).[8] In fact, whenever the additional regularization term is a quadratic form Q_L, we can view the semi-supervised algorithm as finding the optimal function in a data-dependent "warped" RKHS (see Sindhwani et al. (2005) for details).

While we certainly agree that Problem 19 gives a finite-dimensional solution to an infinite-dimensional problem, we question whether this problem is the "right" problem to solve. In particular, we consider two "strategies" for classifying new points:

8. We have not seen many examples in the machine learning literature of researchers treating the outputs y as the predictive values. The above algorithms are generally phrased in terms of c, or sometimes in terms of $f(x)$, with $f(x)$ viewed as dependent on c.

- **(Inductive)** Solve Problem 19. Compute c via $y = Kc$, yielding a "function" $f(x) = \sum_i c_i K(x, x_i)$. Use f to predict values at new points.

- **(Transductive)** When presented with a new point, compute an augmented form of optimization Problem 19 by adding the new point to the RKHS and additional regularization terms. Predict the value at the new point as the associated optimal value in the augmented optimization problem.

Given sufficient computational resources, we consider the transductive approach to be the "natural" choice, in that it treats previous and future unlabelled points on an equal basis. The inductive algorithm may be appropriate as a computationally more tractable substitute, but it is crucial to keep in mind that we do not expect the inductive and transductive approaches to give the same answer. This is in direct contrast to the case of standard RKHS supervised learning, where Theorem 25 shows that the inductive and transductive approaches *give the same answer*. For RKHS supervised learning, unlabelled points do not contribute to the loss (we implicitly assume $v_i(y_i) = 0$ for unlabelled points). On the other hand, when we consider a data-dependent smoothness functional over unlabelled data, additional points give us additional information about this smoothness penalizer. In the transductive semi-supervised case, adding the unlabelled point to the optimization problem can change the predictions at previous points, both labelled and unlabelled. We will see another example of this phenomenon in Section 8.

An alternate perspective on problems like Problem 19 is to think of the smoothness penalty $Q_L(y)$ as part of the regularizer, rather than part of the loss. From this viewpoint, we no longer have a (data independent) RKHS regularizer, and so Theorem 25 does not apply.

8. Learning the Kernel

Standard RKHS regularization involves a fixed kernel function k and kernel matrix K. Recently, there has been substantial interest in scenarios in which a kernel matrix or kernel function is learned from the data, simultaneously with learning a prediction function. We first develop some very general results on this case, and then show how these results can easily be used to obtain and generalize results from the recent literature.

We consider a variant of Tikhonov regularization where the regularization term itself contains parameters $u \in \mathbb{R}^p$, which are optimized; we learn the regularization in tandem with the regression. The primal becomes

$$\inf_{y,u} \left\{ R(y,u) + \sum_i v_i(y) \right\}.$$

where $R(y,u)$ is convex in y. Given the independence of the loss from u, we can move the u-infimum inside

$$\inf_{y} \left\{ \inf_{u} R(y,u) + \sum_i v_i(y) \right\}$$

obtaining a new regularization $R'(y) = \inf_u R(y,u)$. We will assume through the remainder that $R(y,u)$ is ccp.

By lemma 11, $R'^*(z) = R^*(z,0)$. If R' is exact then the optimality condition for the regularization becomes

$$R(y,u) - y^t z + R^*(z,0) = 0,$$

or, equivalently, $(z,0) \in \partial R(y,u)$.

We now specialize to the RKHS case, and allow the entire kernel to play the role of the auxiliary variables: $R(y,K) = \lambda Q_{K^{-1}}(y) + F(K)$ for some closed convex F whose domain $\mathcal{K}$ is contained in the $n \times n$ symmetric positive semidefinite matrices. Let $A \bullet B \equiv \sum_{i,j} A_{ij} B_{ij}$ and recall $zz^t \bullet K = z^t K z$. R is ccp and $R^*(z,W) = F^*\left(W + \frac{1}{2}\lambda^{-1} zz^t\right)$ which we can verify by Theorem 5, taking the conjugate

$$\begin{aligned}
R^*(z,W) &= \sup_{y,K} \left\{ y^t z + K \bullet W - \lambda Q_{K^{-1}}(y) - F(K) \right\} \\
&= \sup_{K} \left\{ \sup_{y} \{ y^t z - \lambda Q_{K^{-1}}(y) \} + K \bullet W - F(K) \right\} \\
&= \sup_{K} \left\{ \lambda^{-1} Q_K(z) + K \bullet W - F(K) \right\} \\
&= \sup_{K} \left\{ K \bullet \left(W + \frac{1}{2}\lambda^{-1} zz^t \right) - F(K) \right\} \\
&= F^* \left(W + \frac{1}{2}\lambda^{-1} zz^t \right)
\end{aligned}$$

and biconjugate,

$$\begin{aligned}
\sup_{z,W} \left\{ y^t z + K \bullet W - F^* \left(W + \frac{1}{2}\lambda^{-1} zz^t \right) \right\} &= \\
\sup_{z} \left\{ y^t z + \sup_{W} \left\{ K \bullet W - F^* \left(W + \frac{1}{2}\lambda^{-1} zz^t \right) \right\} \right\} &= \\
\sup_{z} \left\{ y^t z - K \bullet \frac{1}{2}\lambda^{-1} zz^t \right\} + F^{**}(K) &= \\
\sup_{z} \left\{ y^t z - \lambda^{-1} Q_K(z) \right\} + F^{**}(K) &= \\
\lambda Q_{K^{-1}}(y) + F(K) &= R(y,K).
\end{aligned}$$

By Lemmas 11 and 13, $R'^* = F^*\left(\frac{1}{2}\lambda^{-1} zz^t\right)$ and R' is closed and exact if $\forall z \in \mathbb{R}^n, zz^t \in \operatorname{int}(\operatorname{dom} F^*)$. If R' is exact, the condition for optimality for the regularization (i.e., $z \in \partial R'(y)$) is

$$\frac{1}{2}\lambda y^t K^{-1} y + F(K) - y^t z + F^* \left(\frac{1}{2}\lambda^{-1} zz^t \right) = 0$$

$$\left[\frac{1}{2}\lambda y^t K^{-1} y - y^t z + \frac{1}{2}\lambda^{-1} z^t K z \right] + \left[F(K) - \operatorname{tr}\left(\frac{1}{2}\lambda zz^t K \right) + F^* \left(\frac{1}{2}\lambda^{-1} zz^t \right) \right] = 0.$$

Each of the two bracketed terms is non-negative, by Theorem 6; therefore they must both vanish. Hence, $y = \lambda^{-1} K z$ and $\frac{1}{2}\lambda^{-1} zz^t \in \partial F(K)$, equivalently, $K \in \partial F^*\left(\frac{1}{2}\lambda^{-1} zz^t\right)$.

Given an RKHS regularizer, learning K and y simultaneously will be a convex problem as long as F is closed and convex. However, in order to obtain a useful learning algorithm, the function

F must provide a meaningful constraint on allowable kernels, and there must be a mechanism for predicting values at new points. Abusing notation, let F_N and F_M be the versions of F that operate on $n \times n$ and $m \times m$ matrices, respectively (we usually suppress this notation). Suppose that, whenever $m > n$, we have the property that $F_N(A) = \inf_{K:K_{NN}=A} F_M(K)$. In this case, it is straightforward to see that adding additional "testing" points to the regularizer, but not the loss, will not change the objective value, nor will it change the y values at the "training" points. This leads to a transductive algorithm. Furthermore, if the minimizer in the inf is unique, we can use K_N to determine K_M given the new x points, we will have a representer theorem, and the inductive and transductive cases will be identical.

Lanckriet et al. (2004) consider a transductive scenario, where the training and testing points are known in advance. They start with a finite set of kernels $K_1, \dots, K_k$ (the K_i are over the training and testing sets) and take $F(K) = \delta_{K_{u,c}}(K)$, where

$$K_{u,c} \equiv \left\{ K : K = \sum_i u_i K_i, \mathrm{tr}(K) = c,\ K \succeq 0 \right\}.$$

(Lanckriet et al. (2004) frequently consider $K_{u^+,c}$ as well, where the u are constrained to be nonnegative.) A primary concern in Lanckriet et al. (2004) is showing that when the SVM hinge loss is used, the resulting optimizations can be phrased as semidefinite programming problems. They recognize that it is important to "entangle" the training and testing kernel matrices—if we simply allowed K to range over the entire semidefinite cone, for example, we would obtain kernel matrices which fit the training data well and ignored the testing data. Using a finite combination of kernel matrices essentially means that we are learning a kernel function parametrized by u, and applying this function to both training and testing points. Because Lanckriet et al. (2004) constrain the trace of the *entire* kernel matrix K, adding additional testing points can change the value at training points, and they cannot easily and directly perform induction. If they had instead chosen to constrain the trace of the kernel matrix over only the training points, or simply to constrain the sum of the u_i, they could have obtained a representer theorem, and there would have been agreement between their transductive algorithm and the obvious inductive algorithm. Lanckriet et al. (2004) focus primarily on the SVM hinge loss. In fact, any convex loss can be used, and the resulting optimization problem can be cast as a semidefinite programming problem (Recht, 2006).

While Lanckriet et al. (2004) consider a finitely generated set of kernel matrices, Argyriou et al. (2005) works with a convex set generated by an infinite, continuously parametrized set of kernel functions, the primary example being Gaussian kernels with all bandwidths in some closed interval. Argyriou et al. (2005) show that the optimal kernel function will have a "representation" in terms of $n+1$ basic kernels. Because we have separated our concerns with the kernel function appearing only in the regularization and not in the loss, we are able to give a very simple proof of their result. Given a subset, $\mathcal{K}$, of the $n \times n$ symmetric positive semidefinite matrices, we consider $F(K) = \delta_{\mathcal{K}^\oplus}(K)$, where $\mathcal{K}^\oplus$ is the convex hull of $\mathcal{K}$.

Lemma 26 *Let $z \in \mathbb{R}^n$ and $K \in \mathcal{K}^\oplus$. There exists $\tilde{K} = \sum_{i=1}^{n+1} t_i K_i$ with $t_i \geq 0$ and $\sum_i t_i = 1$ such that $\tilde{K}z = Kz$.*

Proof Let $Y = \{K'z : K' \in \mathcal{K}\}$ and $Y^\oplus = \{K'z : K' \in \mathcal{K}^\oplus\}$. Clearly, $Y^\oplus$ is the convex hull of Y and $Kz \in Y^\oplus$. By Carathéodory's theorem (Rockafellar and Wets (2004), Theorem 2.29), $\exists t \in \mathbb{R}^{n+1}, t_i \geq 0, \sum_i t_i = 1$ such that $Kz = \sum_i t_i y_i$ with $y_i \in Y$ and hence $y_i = K_i z$ for some $K_i \in \mathcal{K}$. ■

$F^* = \sigma_{\mathcal{K}^\oplus}$. Let us assume that $\mathcal{K}^\oplus$ is closed (compactness of $\mathcal{K}$ is sufficient but not necessary). Under this assumption, F is ccp, and the optimality condition from the regularization term becomes

$$y = \lambda^{-1}Kz \quad \text{and} \quad z^tKz = \sigma_{\mathcal{K}^\oplus}(zz^t), K \in \mathcal{K}^\oplus. \tag{20}$$

Corollary 27 *Let $z \in \mathbb{R}^n, K \in \mathcal{K}^\oplus$ and $\tilde{K} \in \mathcal{K}^\oplus$ be as in Lemma 26. If y, z, K satisfy (20) then $y, z, \tilde{K}$ do. Additionally, if $t_i > 0$ then $z^tK_iz = \sigma_{\mathcal{K}^\oplus}(zz^t)$.*

Proof The first claim comes directly from $\tilde{K}z = Kz$. The second claim comes from $z^t\tilde{K}z = \sigma_{\mathcal{K}^\oplus}(zz^t)$ and a standard argument. ■

Since K does not appear in the loss optimality conditions (which depend only on z and y), we see that we can construct an optimal y, K, and z for the entire kernel learning problem where K is a convex combination of at most $n+1$ "atoms" of $\mathcal{K}$ that solve the problem. Our result is both substantially simpler than the result of Argyriou et al. (2005), and more general in that it applies to arbitrary convex loss functions: Argyriou et al. (2005)'s argument was essentially a saddle point argument which required differentiability of the loss.

Our work generalizes previous results in this field, in that we have shown that one may use an arbitrary kernel penalizer $F(K)$; previous work has used only δ functions. Exploring alternate penalizers and developing algorithms is a topic for future work.

9. Regularizations and Bias

In this section, we consider relaxing primal regularizations by means of infimal convolutions. The primary application is unregularized *bias* terms for kernel machines. In Section 9.1, we obtain very general results for biased regularizations, independent of particular loss functions and even particular regularizers. In Section 9.2, we use similar techniques to analyze leave-one-out computations for RKHS learning.

9.1 Learning with Bias

So far, we have considered regularization in an RKHS, deriving formulations of regularized least squares and an unbiased version of the support vector machine. In practice, the SVM is generally used with an unregularized *bias* term b (Poggio et al., 2001). This gives the regularization the property that $R(y + b1_n) = R(y)$—constant shifts of the regression values are free. This can be achieved by taking a base regularization such as $R = \lambda Q_{K^{-1}}$ and replacing it with $R' = \lambda Q_{K^{-1}} \star \delta_{1_n\mathbb{R}}$ where $1_n \in \mathbb{R}^n$ is the vector of all 1s. In this case, we will see that the subgradient relation for the regularization becomes

$$z \in \partial R'(y) \Leftrightarrow \begin{pmatrix} \lambda^{-1}K & 1_n \\ 1_n^t & 0 \end{pmatrix} \begin{pmatrix} z \\ b \end{pmatrix} = \begin{pmatrix} y \\ 0 \end{pmatrix}, \tag{21}$$

which relates y to both z and the bias parameter b.

We examine biased regularizations more generally. Without loss of generality, we cast this theorem in terms of an infimal convolution in the primal regularization, though by biconjugation, this lemma applies symmetrically to the primal and dual.

Lemma 28 (bias) *Let $C \subset \mathbb{R}^n$ be a closed convex set containing* 0. *Let $R : \mathbb{R}^n \to (-\infty, \infty]$ be ccp where $R' = R \star \delta_C$ is exact. Then $z \in \partial R'(y)$ iff $\exists c \in C$ such that*

$$\begin{aligned} \forall c' \in C, z^t(c'-c) &\leq 0 \\ R(y-c) - (y-c)^t z + R^*(z) &= 0. \end{aligned} \tag{22}$$

Proof By Theorem 15, $z \in \partial R'(y)$ iff $\exists c$ such that $z \in \partial \delta_C(c)$ and $z \in \partial R(y-c)$. The first condition is given by Equation 13, and the second condition is the generic condition of Theorem 6. ■
The first condition in Lemma 28 is a normality condition on z taking different forms depending on the geometry of C. It states that the hyperplane given by $H_{z,\sigma_C(z)}(y) = 0$ is a supporting hyperplane of C. For example, if the boundary of C were smooth, then this condition reduces to z being a normal to C at c. If $C = \mathbb{B}$, then $z = \lambda c$ for some $\lambda \geq 0$.

If C is a vector subspace, then $\sigma_C = \delta_{C^\perp}$, and (22) becomes the condition $z \in C^\perp$ with c an arbitrary element of C. Thus, it seems worthwhile to specialize to the case of vector subspaces.

Corollary 29 *Let $V_1 \subset V_2 \subset \mathbb{R}^n$ be vector subspaces. Let $R : \mathbb{R}^n \to (-\infty, \infty]$ be ccp. Let $R' = R \star \delta_{V_1} + \delta_{V_2}$. Then $R'^* = R^* \star \delta_{V_2^\perp} + \delta_{V_1^\perp}$ and (assuming exactness of the $\star$'s) $z \in \partial R'(y)$ iff $y \in V_2, z \in V_1^\perp$ and $z - a \in \partial R(y-b)$ for some $a \in V_2^\perp, b \in V_1$.*

Proof Let $R_1 = R \star \delta_{V_1}$ and $R_2 = R_1 + \delta_{V_2}$. By Lemma 28,

$$\begin{aligned} z_1 \in \partial R_1(y_1) &\Leftrightarrow z_1 \in V_1^\perp \text{ and } \exists b \in V_1 \text{ s.t. } z_1 \in \partial R_1(y_1 - b) \\ z_2 \in \partial R_2(y_2) &\Leftrightarrow y_2 \in V_2 \text{ and } \exists a \in V_2^\perp \text{ s.t. } z_2 - a \in \partial R_1(y_2). \end{aligned}$$

Thus,

$$\begin{aligned} z \in \partial R'^*(y) \quad &\Leftrightarrow \quad y \in V_2 \text{ and } \exists a \in V_2^\perp \text{ s.t. } z - a \in V_1^\perp \\ &\qquad \text{and } \exists b \in V_1 \text{ s.t. } z - a \in \partial R(y-b) \\ &\Leftrightarrow \quad y \in V_2 \text{ and } z \in V_1^\perp \text{ and } \exists a \in V_2^\perp, b \in V_1 \text{ s.t. } z - a \in \partial R(y-b), \end{aligned}$$

where the last line is due to $V_2^\perp \subset V_1^\perp$. ■

Suppose we start with a regularization R, and we wish to allow a free constant bias in the y values. Our primal problem is:

$$\inf_{y,b} \left\{ R(y - b1_n) + \sum_{i=1}^n (1 - y_i Y_i)_+ \right\} = \inf_y \left\{ (R \star \delta_{1_n \mathbb{R}})(y) + \sum_{i=1}^n (1 - y_i Y_i)_+ \right\}.$$

In the dual, the regularization term becomes $(R^* + \delta_{(1_n\mathbb{R})^\perp})(z)$. We see that the dual regularizer $z \in (1_n\mathbb{R})^\perp$ can also be written as the constraint $\sum z_i = 0$. We have shown that for *any* regularization and loss function, allowing a free unregularized constant b in the y values induces a constraint $\sum_i z_i = 0$ in the dual problem. *Nothing else changes:* to obtain the standard "biased" SVM instead of the unbiased SVM we derived in Section 5.2.2, we change the primal regularizer from $Q_{K^{-1}}$ to $Q_{K^{-1}} \star \delta_{1_n \mathbb{R}}$, we change the dual regularizer from Q_K to $Q_K + \delta_{(1_n\mathbb{R})^\perp}$, or, equivalently, we add the constraint $\sum_i z_i = 0$ to the dual optimization problem. There is no need to take the entire dual again—the loss function is unchanged, and in the Fenchel formulation, the regularization and loss make separate contributions.

In terms of Corollary 29, the standard constant bias is obtained by choosing $V_1 = 1_n\mathbb{R}, V_2 = \mathbb{R}^n$, and therefore $a = 0$ and $R \star \delta_{V_1} + \delta_{V_2} = R \star \delta_{V_1}$. Assuming we are using an RKHS primal regularizer $Q_{K^{-1}}$, then $z \in \partial(Q_{K^{-1}} \star \delta_{V_1})(y-b)$ iff $b \in 1_n\mathbb{R}$ and $z \in \partial Q_{K^{-1}}(y-b)$, or, equivalently, if (21) holds.

We can incorporate the same bias into regularized least squares. Recalling that the loss optimality condition for RLS is $y+z = Y$, we can eliminate either y or z from the optimality condition, obtaining either a pure primal or pure dual formulation for the so-called *least squares SVM* (Suykens et al., 2002).

$$\begin{aligned}
\begin{pmatrix} \lambda^{-1}K & 1 \\ 1^t & 0 \end{pmatrix}\begin{pmatrix} Y_i \\ b \end{pmatrix} &= \begin{pmatrix} I+\lambda^{-1}K & 1 \\ 1^t & 0 \end{pmatrix}\begin{pmatrix} y \\ 0 \end{pmatrix} \\
\begin{pmatrix} I+\lambda^{-1}K & 1 \\ 1^t & 0 \end{pmatrix}\begin{pmatrix} z \\ b \end{pmatrix} &= \begin{pmatrix} Y_i \\ 0 \end{pmatrix},
\end{aligned}$$

Note that introducing biases via infimal convolutions with RKHS regularization preserves the representer property:

Corollary 30 *Let $K = \begin{pmatrix} K_{NN} & K_{NM} \\ K_{MN} & K_{MM} \end{pmatrix} \in \mathbb{R}^{(n+m)\times(n+m)}$ be symmetric positive definite where $K_{NN} \in \mathbb{R}^{n\times n}$. Let $C \in \mathbb{R}^{n+m}$ be a closed convex set containing 0, and let C_N be the projection of C onto the first N dimensions. For all $y_1 \in \mathbb{R}^n$,*

$$\inf_{y_2\in\mathbb{R}^m} (Q_{K^{-1}} \star \delta_C)(y_1,y_2) = (Q_{K_{NN}^{-1}} \star \delta_{C_N})(y_1),$$

where the minimizer is of the form $y_2 = K_{MN}K_{NN}^{-1}(y_1-c_1)+c_2$ where $(c_1,c_2) \in C$.

Proof

$$\begin{aligned}
\inf_{y_2\in\mathbb{R}^m} (Q_{K^{-1}} \star \delta_C)(y_1,y_2) &= \inf_{y_2\in\mathbb{R}^m} \inf_{(c_1,c_2)\in C} Q_{K^{-1}}((y_1,y_2)-(c_1,c_2)) \\
&= \inf_{(c_1,c_2)\in C} \inf_{y_2\in\mathbb{R}^m} Q_{K^{-1}}((y_1,y_2)-(c_1,c_2)) \\
&= \inf_{c_1\in C_N} \inf_{y_2'\in\mathbb{R}^m} Q_{K^{-1}}(y_1-c_1, y_2') \\
&= \inf_{c_1\in C_N} Q_{K_{NN}^{-1}}(y_1-c_1),
\end{aligned}$$

where the last equality is an application of Corollary 24 (the statement about y_2 then follows since that $Q_{K^{-1}} \star \delta_C$ and $Q_{K_{NN}^{-1}} \star \delta_{C_N}$ are exact). ■

The minimizing y_2 will be unique if a *transversality* condition holds: $\forall (c_1,c_2),(c_1',c_2') \in C, c_1 = c_1' \Rightarrow c_1 = c_2'$. For the standard constant-term bias, this is clear and the recovery of y_2 reduces to $y_2 = K_{MN}K_{NN}^{-1}(y_1 - 1_nb) + 1_mb$.

9.2 Leave-one-out Computations

For small data sets, it is frequently of interest to do model selection via leave-one-out cross validation. Done naively, this involves solving n optimization problems, one for each size $n-1$ subset of the data, and testing $n-1$ functions at the remaining points. In this section, we relate the solution of the full problem (using all the data points) to the leave-one-out value.

We consider an $n+1$ point data set, and define $y = (y_0, y_N) \in \mathbb{R}^{n+1}$ in a regression problem with losses $v_i(y)$ for $0 \le i \le n$. The "full" optimization problem, with an RKHS regularization, is

$$\inf_{y \in \mathbb{R}^{n+1}} \left\{ \lambda Q_{K^{-1}}(y) + \sum_{i=0}^{n} v_i(y_i) \right\}.$$

We now show how we can compute a "leave one out" score as the auxiliary variable of a biased regularization of the form δ_C where $C = \{(y_0, 0) \in \mathbb{R}^{n+1}\}$. Consider the optimization,

$$\inf_{y \in \mathbb{R}^{n+1}} \left\{ \lambda (Q_{K^{-1}} \star \delta_C)(y) + \sum_{i=0}^{n} v_i(y_i) \right\} = \inf_{y \in \mathbb{R}^{n+1}, \beta \in \mathbb{R}} \left\{ \lambda Q_{K^{-1}}(y - \beta e_0) + \sum_{i=0}^{n} v_i(y_i) \right\}.$$

By Corollary 29, the right hand problem has the regularization optimality condition

$$\begin{pmatrix} \lambda^{-1}K & e_0 \\ e_0^t & 0 \end{pmatrix} \begin{pmatrix} z \\ \beta \end{pmatrix} = \begin{pmatrix} y \\ 0 \end{pmatrix},$$

Clearly, at optimality, $z_0 = 0$ and y_0 will assume a value with minimal loss: $v_0(y_0) = \inf_y v(y) \equiv \bar{v}$. With $z_0 = 0$, it is clear that the values $z_1, \ldots, z_n$ and $y_1, \ldots, y_n$ are an optimal solution to the restricted problem obtained by simply discarding (X_0, Y_0), and that therefore, $(\lambda^{-1}Kz)_0$ is the value we would obtain if we solved the restricted problem and computed the "test value" at X_0. β is the leave-one-out "error" $y_0 - (\lambda^{-1}Kz)_0$.

In the case of a square loss, the loss optimality conditions are $z + y = Y$, allowing us to eliminate y,

$$\begin{pmatrix} I + \lambda^{-1}K & e_0 \\ e_0^t & 0 \end{pmatrix} \begin{pmatrix} z \\ \beta \end{pmatrix} = \begin{pmatrix} Y \\ 0 \end{pmatrix},$$

and solve for β, Defining $G \equiv I + \lambda^{-1}K$, we have

$$\beta \;=\; \frac{e_0^t G^{-1} Y}{e_0^t G^{-1} e_0},$$

the standard formula for RLS leave-one-out errors.

10. Low-rank Approximation of Kernel Matrices

Given n data points, the kernel matrix K is n by n. For large n, it is impractical to compute with (or store) K. Consequently, there has been an interest in approximations to K. In this section, we consider a class of low rank approximations to K obtained by considering a vector subspace $V_0 \subset \mathbb{R}^n$, and defining $\tilde{K}$ as the lowest rank symmetric matrix such that $\tilde{K}V_0 = KV_0$. The primary example is the well-known Nyström approximation (Baker, 1977). When the Nyström kernel matrix approximation was first used in machine learning applications (Willams and Seeger, 2000), the suggested approach was to simply replace K with $\tilde{K}$; we refer to this approach as the *Nyström method*. We will show that the "natural" algorithm suggested by the Nyström *approximation* is in fact the *subset of regressors* method (Poggio and Girosi, 1990; Luo and Wahba, 1997), as opposed to the Nyström method.[9] In contrast, the Nyström method essentially makes an unwarranted (and incorrect) assumption that the representer theorem, which directly connects the dual variables z and the function expansion coefficients c in the standard RKHS case, still holds when we replace K with $\tilde{K}$. Our observation is consonant with empirical reports (Rifkin, 2002; Rasmusen and Williams, 2006) that the subset of regressors algorithm tends to outperform the Nyström algorithm. We note in passing that whereas previous algorithmic work with the Nyström approximation has mostly focussed on Gaussian Process regression or regularized least-squares, our results are all independent of the particular loss function chosen.

We first show that $Q_{\tilde{K}}$ has a simple characterization.

Lemma 31 *Let $K, \tilde{K} \in \mathbb{R}^{n\times n}$ with K symmetric positive definite and $\tilde{K}V_0 = KV_0$ with $\operatorname{rank}(\tilde{K}) = \dim V_0$. Then $Q_{\tilde{K}} = Q_K \star \delta_{(KV_0)^\perp}$.*

Proof For all $v \in V_0$ and $n \in Null(\tilde{K})$, $0 = n^t\tilde{K}v = n^tKv$, thus, since $\operatorname{rank}(\tilde{K}) = \dim V_0$, $Null(\tilde{K}) = (KV_0)^\perp$ and $V_0 \cap (KV_0)^\perp = \{0\}$.

Given $v \in V_0, n \in (KV_0)^\perp$,

$$\begin{aligned} Q_K(v+n) &= \frac{1}{2}(v+n)^tK(v+n) \\ &= Q_K(v) + Q_K(n) \\ Q_{\tilde{K}}(v+n) &= (v+n)^t\tilde{K}(v+n) \\ &= \frac{1}{2}v^t\tilde{K}v = \frac{1}{2}v^tKv = Q_K(v). \end{aligned}$$

In comparison

$$\begin{aligned} (Q_K \star \delta_{(KV_0)^\perp})(v+n) &= \inf_w \left\{ Q_K(v+n-w) + \delta_{(KV_0)^\perp}(w) \right\} \\ &= Q_K(v) + \inf_{w\in(KV_0)^\perp} \{Q_K(n-w)\} \\ &= Q_K(v). \end{aligned}$$

■

Lemma 31 implies

$$\begin{aligned} \inf_z \{\lambda^{-1}Q_{\tilde{K}}(z) + V^*(-z)\} &= \inf_z \left\{\lambda^{-1}(Q_K \star \delta_{(KV_0)^\perp})(z) + V^*(-z)\right\} \\ &= \inf_{z\in\mathbb{R}^n, z'\in(KV_0)^\perp} \{\lambda^{-1}Q_K(z-z') + V^*(-z)\} \\ &= \inf_{z''\in\mathbb{R}^n} \left\{\lambda^{-1}Q_K(z'') + \inf_{z'\in(KV_0)^\perp} V^*(-z''-z')\right\}. \end{aligned} \tag{23}$$

We see that giving the regularization a non-trivial nullspace ($(KV_0)^\perp$) is equivalent to allowing an unregularized "bias" from that same space. Hence, we expect the resulting modified dual optimization problem to have a lower value than the original dual.

9. To avoid confusion, we reiterate that the *Nyström approximation* is a low-rank approximation to K, denoted by $\tilde{K}$, while the *Nyström method* is obtained by simply replacing K with $\tilde{K}$ in an algorithm such as regularized least squares.

The modified primal problem, the Fenchel dual of (23), is

$$\inf_y \{\lambda Q_{K^{-1}}(y) + \delta_{KV_0}(y) + V(y)\} \quad = \quad \inf_{y \in KV_0} \{\lambda Q_{K^{-1}}(y) + V(y)\} \tag{24}$$

which is identical to the original optimization problem, but with a restricted domain, and hence a higher value, in general. This is to be expected, as any pair of Fenchel duals have inversely related infimal values. For the remainder of this section, the variables y and z refer, respectively to optimal solutions to the modified primal (24) and dual (23) problems.

So far, we have considered an arbitrary subspace V_0. We now specialize to the Nyström approximation, obtained by partitioning the data into two blocks N and M with $K = \begin{pmatrix} K_{NN} & K_{NM} \\ K_{MN} & K_{MM} \end{pmatrix}$, and taking $V_0 = \begin{pmatrix} I \\ 0 \end{pmatrix} \mathbb{R}^n$, in which case

$$\tilde{K} = \begin{pmatrix} K_{NN} & K_{NM} \\ K_{MN} & K_{MN}K_{NN}^{-1}K_{NM} \end{pmatrix} = \begin{pmatrix} I & 0 \\ K_{MN}K_{NN}^{-1} & 0 \end{pmatrix} \begin{pmatrix} K_{NN} & K_{NM} \\ K_{MN} & K_{MM} \end{pmatrix}.$$

It is obvious that $\tilde{K}V_0 = KV_0$ and the expression on the right demonstrates that $\tilde{K}$ is rank n.

By Corollary 29, with $V_2 = K \begin{pmatrix} I \\ 0 \end{pmatrix} \mathbb{R}^n = \left(K^{-1} \begin{pmatrix} 0 \\ I \end{pmatrix} \mathbb{R}^m\right)^{\perp}$ and $V_1 = \{0\}$, the complementarity relation between y and z due to the regularization terms of the modified problems (24) and (23) is

$$\begin{pmatrix} z \\ 0 \end{pmatrix} = \lambda \begin{pmatrix} K^{-1} & K^{-1}\begin{pmatrix} 0 \\ I \end{pmatrix} \\ (0 \quad I)K^{-1} & 0 \end{pmatrix} \begin{pmatrix} y \\ \alpha \end{pmatrix}$$

or, equivalently, $\exists r \in \mathbb{R}^n$ such that

$$y \quad = \quad \lambda^{-1} K \begin{pmatrix} r \\ 0 \end{pmatrix} \tag{25}$$

and

$$\lambda^{-1}Kz \quad = \quad y + \begin{pmatrix} 0 \\ \alpha \end{pmatrix} \tag{26}$$

$$\lambda^{-1}\tilde{K}z \quad = \quad y, \tag{27}$$

where (27) is obtained from (26) by multiplying the first equation on the left by $\begin{pmatrix} I & 0 \\ K_{MN}K_{NN}^{-1} & 0 \end{pmatrix}$ and observing that $\tilde{K}\begin{pmatrix} r \\ 0 \end{pmatrix} = K\begin{pmatrix} r \\ 0 \end{pmatrix}$ by construction.

In standard Tikhonov regularization in an RKHS, the complementarity equation from the regularization, $y = \lambda^{-1}Kz$, and the representer theorem tells us that we can obtain an optimizing function using $c = \lambda^{-1}z$. Once we have replaced K with $\tilde{K}$, this connection no longer holds, and we must decide how to generalize a minimizer of (23) or (24) into a function that can be used to predict the values at future points. The Nyström method, suggested in Willams and Seeger (2000) is to use the function $f_d(x) \equiv \sum_{i=1}^{n+m} \lambda^{-1} z_i k(x, X_i)$, essentially "pretending" that the connection between c

and z is still valid. An alternate choice is to use r, yielding $f_p(x) \equiv \sum_{i=1}^{n} \lambda^{-1} r_i k(x, X_i)$.[10] By (26), $f_p(X_i) = f_d(X_i)$ when $1 \leq i \leq n$, while $f_d(X_i) = f_p(X_i) + \alpha_{i-n}$ when $n < i \leq n+m$—the two regressors are related, taking the same values at the first n points but generally different values on the last m points.

By eliminating y for r in the modified primal problem (24), we see that f_p is the solution of the *subset of regressors* method, where we construct a function using only those coefficients associated with points in N:

$$\inf_{y \in KV_0} \{\lambda Q_{K^{-1}}(y) + V(y)\} \quad = \quad \inf_{r \in \mathbb{R}^n} \left\{ \lambda^{-1} Q_{K_{NN}}(r) + V\left(\lambda^{-1} \begin{pmatrix} K_{NN} \\ K_{MN} \end{pmatrix} r \right) \right\}.$$

In this sense, we can see that the subset of regressors method is associated with the modified problems (24) and (23). Additionally, the function f_p will recover the y values optimizing (24) at all $n+m$ training points. In contrast, if we use the Nyström method on the last m training points, we will *not* recover the last m values—they will differ by α.

We have shown that the subset of regressors method corresponds in a natural way to the modified primal and dual problems obtained by replacing K with the Nyström approximation $\tilde{K}$, whereas the Nyström method makes the additional unwarranted assumption that it is still a good idea to construct a function to classify future points using $c = \lambda^{-1} z$. We can also derive an interesting relationship between the functions f_p and f_d.

We first note that by (25), y and $\begin{pmatrix} 0 \\ \alpha \end{pmatrix}$ are "K^{-1} orthogonal": $y^t K^{-1} \begin{pmatrix} 0 \\ \alpha \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix}^t \begin{pmatrix} 0 \\ \alpha \end{pmatrix} = 0$. As a consequence, the quadratic form $Q_{K^{-1}}$ "distributes" over y and $\begin{pmatrix} 0 \\ \alpha \end{pmatrix}$:

$$\begin{aligned} Q_{K^{-1}}(\lambda^{-1} K z) &= Q_{K^{-1}}\left(y + \begin{pmatrix} 0 \\ \alpha \end{pmatrix} \right) \\ &= Q_{K^{-1}}(y) + y^t K^{-1} \begin{pmatrix} 0 \\ \alpha \end{pmatrix} + Q_{K^{-1}} \begin{pmatrix} 0 \\ \alpha \end{pmatrix} \\ &= Q_{K^{-1}}(y) + Q_{K^{-1}} \begin{pmatrix} 0 \\ \alpha \end{pmatrix}. \end{aligned}$$

Furthermore, by (26),

$$\begin{aligned} \begin{pmatrix} 0 \\ \alpha \end{pmatrix}^t z &= \begin{pmatrix} 0 \\ \alpha \end{pmatrix}^t \left(\begin{pmatrix} r \\ 0 \end{pmatrix} + \lambda K^{-1} \begin{pmatrix} 0 \\ \alpha \end{pmatrix} \right) \\ &= \lambda \begin{pmatrix} 0 \\ \alpha \end{pmatrix}^t K^{-1} \begin{pmatrix} 0 \\ \alpha \end{pmatrix} \\ &= 2\lambda Q_{K^{-1}} \begin{pmatrix} 0 \\ \alpha \end{pmatrix}. \end{aligned}$$

We now consider comparing the functions f_d and f_p. Recall that the values of f_d at all training points are equal to $\lambda^{-1} K z$ while those of f_p are y. Because y and z are optimal solutions to the modified

10. We chose the name f_d and f_p because the two functions seem to us "suggested" by the modified dual and primal problems, respectively.

primal and dual problems, $-z \in \partial V(y)$. Since $-z$ is a subgradient of V at y, $V(y) - z^t(y' - y) \leq V(y')$ for all $y' \in \mathbb{R}^n$. Setting $y' = \lambda^{-1}Kz$, and noting that $\lambda^{-1}Kz - y = \begin{pmatrix} 0 \\ \alpha \end{pmatrix}$, by (26), we have,

$$\begin{aligned} V(y) - V(y') \leq z^t(y' - y) &= z^t \begin{pmatrix} 0 \\ \alpha \end{pmatrix} = 2\lambda \left(Q_{K^{-1}}(\lambda^{-1}Kz - y) \right) \\ &= 2\lambda \left(Q_{K^{-1}}(y') - Q_{K^{-1}}(y) \right) = 2\lambda Q_{K^{-1}} \begin{pmatrix} 0 \\ \alpha \end{pmatrix}. \end{aligned}$$

Recall that y was obtained by minimizing $V(y) + \lambda Q_{K^{-1}}(y)$ over KV_0. Comparing the function values y and y' of the regressors f_p and f_d respectively, we see that f_d pays a higher regularization cost as compared to f_p, $Q_{K^{-1}}(y') - Q_{K^{-1}}(y) = Q_{K^{-1}} \begin{pmatrix} 0 \\ \alpha \end{pmatrix}$. Furthermore, the difference in loss at the last m points in favor of f_d is bounded by 2λ times this difference—it is not possible for f_d to have arbitrarily smaller loss. In practice, we often find that the Nyström method produces *worse* function values at the m points than the subset of regressors does. This provides further evidence that the subset of regressors method should be preferred to the Nyström method.

11. Discussion and Future Work

We have introduced a new framework for thinking about optimization in learning theory by combining two key elements: Fenchel duality and value regularization. Fenchel duality allows us to analyze the optimality conditions for regularization and loss terms separately, and combine these optimality conditions to get complete optimality conditions for machine learning schemes. Value regularization makes it easy to reason about optimization problems over training and testing points simultaneously, and isolates the RKHS kernel to the regularization term. We have used the framework to gain new insights into several topics in machine learning, including the representer theorem, learning a kernel matrix, biased regularizations, and low-rank approximations to kernel matrices. There remain several interesting open questions.

In Sections 7 and 8, we showed that both supervised learning in an RKHS and learning the kernel matrix had a representer theorem that caused the inductive and natural transductive algorithms to make the same prediction. It would be interesting to explore this idea further, characterizing formally what sorts of conditions give rise to agreement between inductive and transductive algorithms.

The framework developed in this article is specialized to the case where the predicted outputs y are real-valued scalars. It is frequently of interest to make predictions on objects with more structure: examples include multiclass classification, ranking, and classification of sequences. The extension of the value regularization framework to these problems is in principle straightforward, but many details remain to be worked out.

Finally, although this article has focussed on the analysis of existing schemes, one may ask whether a value regularization perspective can lead to the development of new learning algorithms. It seems unlikely that value regularization in an RKHS will lead directly to improved algorithms (for example, a better SVM solver), because the inverse kernel matrix K^{-1} appearing in the primal is a computationally inconvenient object. However, we might imagine designing regularizers for which value-based optimization is computationally effective, such as a transductive algorithm with a regularizer that directly imposes some non-RKHS smoothness over the input space. We are actively working to design and implement such algorithms.

Acknowledgments

We thank Ben Recht, Tomaso Poggio, Rob Freund, Adrian Lewis and Roger Wets for helpful discussions and suggestions. We thank Gustavo Santos and Tim Sears for finding many technical errors in earlier versions of this work. We thank our anonymous reviewers for helpful suggestions.

References

Yasemin Altun and Alex Smola. Unifying divergence minimization and statistical inference via convex duality. In *Proceedings of the 19th Annual Conference on Lwearning Theory*, 2006.

Yasemin Altun, David McAllester, and Mikhail Belkin. Maximum margin semi-supervised learning for structured variables. In *Neural Information Processing Systems*, 2005.

Andreas Argyriou, Charles A. Micchelli, and Massimiliano Pontil. Learning convex combinations of continuously parametrized basic kernels. In *Proceedings of the 18th Annual Conference on Learning Theory*, 2005.

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

Christopher T. H. Baker. *The Numerical Treatment of Integral Equations*. Clarendon press, 1977.

Mokhtar S. Bazaraa, Hanif D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley-Interscience, 2nd edition, 1993.

Mikhail Belkin and Partha Niyogi. Semi-supervised learning on riemannian manifolds. *Machine Learning*, 56:209–239, 2003.

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from examples. Technical Report TR-2004-06, University of Chicago, 2004.

Yoshua Bengio, Jean-Francois Paiement, Pascal Vincent, Olivier Delalleau, Nicolas Le Roux, and Marie Ouimet. Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps and spectral clustering. In *Neural Information Processing Systems*, 2003.

Jonathan M. Borwein and Adrian S. Lewis. *Convex Analysis and Nonlinear Optimization*. CMS Books in Mathematics. Springer, 2000.

Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. Atomic decomposition by basis pursuit. Technical Report 479, Stanford University Department of Statistics, 1995.

Corinna Cortes and Vladimir Vapnik. Support vector networks. *Machine Learning*, 20:1–25, 1995.

Nello Cristianini and John Shaw-Taylor. *An Introduction To Support Vector Machines*. Cambridge University Press, 2000.

Miroslav Dudík and Robert E. Schapire. Maximum entropy distribution estimation with generalized regularization. In *Proceedings of the 19th Annual Conference on Lwearning Theory*, pages 123–138, 2006.

Theodoros Evgeniou, Massimiliano Pontil, and Tomaso Poggio. Regularization networks and support vector machines. *Advances In Computational Mathematics*, 13(1):1–50, 2000.

Federico Girosi. An equivalence between sparse approximation and support vectomachines. *Neural Computation*, 10:1455–1480, 1998.

Tommi S. Jaakkola and David Haussler. Probabilistic kernel regression models. In *Proceedings of the 1999 Conference on AI and Statistics*. Morgan Kaufmann, 1999.

Gert R. G. Lanckriet, Nello Cristianini, Peter Bartlett, Laurent El Ghaoui, and Michael I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:24–72, 2004.

Zhen Luo and Grace Wahba. Hybrid adaptive splines. *Journal of the American Statistical Society*, 92:107–116, 1997.

Charles A. Micchelli. Interpolation of scattered data: distance matrices and conditionally positive functions. *Constructive Approximation*, 2(1):11–22, 1986.

Tomaso Poggio and Federico Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, September 1990.

Tomaso Poggio, Sayan Mukherjee, Ryan M. Rifkin, Alex Rakhlin, and Alessandro Verri. b. In *Proceedings of the Conference on Uncertainty in Geometric Computations*, 2001.

Ali Rahimi, Benjamin Recht, and Trevor Darrell. Learning appearance manifolds from video. In *Computer Vision and Pattern Recognition*, 2005.

Carl Edward Rasmusen and Chris Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Ben Recht. Unpublished phd thesis. 2006.

Ryan M. Rifkin. *Everything Old Is New Again: A Fresh Look at Historical Approaches to Machine Learning*. PhD thesis, Massachusetts Institute of Technology, 2002.

R. Tyrrell Rockafellar and Roger J. B. Wets. *Variational Analysis*. Springer, Berlin, 2004.

Bernhard Schölkopf, Ralf Herbrich, and Alex J. Smola. A generalized representer theorem. In *14th Annual Conference on Computational Learning Theory*, pages 416–426, 2001.

Vikas Sindhwani, Partha Niyogi, and Mikhail Belkin. Beyond the point cloud: from transductive to semi-supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.

Johan A. K. Suykens, Tony Van Gestel, Jos De Brabanter, Bart De Moor, and Joos Vandewalle. *Least Squares Support Vector Machines*. World Scientific, 2002.

Andrei N. Tikhonov and Vasilii Y. Arsenin. *Solutions of Ill-posed problems*. W. H. Winston, Washington D.C., 1977.

Vladimir Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

Grace Wahba. *Spline Models for Observational Data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial & Applied Mathematics, 1990.

Christopher K. I. Willams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems*, 2000.

Ji Zhu, Saharon Rosset, Trevor Hastie, and Rob Tibshirani. 1-norm support vector machines. In *Neural Information Processing Systems*, 2003.

Journal of Machine Learning Research 8 (2007) 481-507 Submitted 11/05; Revised 10/06; Published 3/07

Integrating Naïve Bayes and FOIL*

Niels Landwehr LANDWEHR@INFORMATIK.UNI-FREIBURG.DE
Kristian Kersting KERSTING@INFORMATIK.UNI-FREIBURG.DE
Department of Computer Science
Albert-Ludwigs-University Freiburg
Georges-Köhler-Allee 79
79110 Freiburg, Germany

Luc De Raedt LUC.DERAEDT@CS.KULEUVEN.BE
Department of Computer Science
Katholieke Universiteit Leuven
Celestijnenlaan 200 A
B-3001 Heverlee, Belgium

Editor: Stefan Wrobel

Abstract

A novel relational learning approach that tightly integrates the naïve Bayes learning scheme with the inductive logic programming rule-learner FOIL is presented. In contrast to previous combinations that have employed naïve Bayes only for post-processing the rule sets, the presented approach employs the naïve Bayes criterion to guide its search directly. The proposed technique is implemented in the NFOIL and TFOIL systems, which employ standard naïve Bayes and tree augmented naïve Bayes models respectively. We show that these integrated approaches to probabilistic model and rule learning outperform post-processing approaches. They also yield significantly more accurate models than simple rule learning and are competitive with more sophisticated ILP systems.

Keywords: rule learning, naïve Bayes, statistical relational learning, inductive logic programming

1. Introduction

The study of learning schemes that lie at the intersection of probabilistic and logic or relational learning has received a lot of attention recently (De Raedt and Kersting, 2003; Getoor and Taskar, 2007). Whereas typical approaches upgrade existing probabilistic learning schemes to deal with relational or logical data (such as Probabilistic Relational Models (Getoor et al., 2001) or Stochastic Logic Programs (Muggleton, 1996)), we start from an inductive logic programming system and extend it with a probabilistic model. More specifically, we start with the simplest approaches from both domains, the inductive logic programming system FOIL (Quinlan, 1990) and naïve Bayes, and integrate them in the NFOIL system. As the strong independence assumption of naïve Bayes can be problematic in some domains, we furthermore investigate a generalization of naïve Bayes known as tree augmented naïve Bayes in the TFOIL system. Indeed, this methodology could be extended to learning full Bayesian networks; however, the advantage of combining such simple learning

*. This is a significant extension of a paper that appeared in the Proceedings of the Twentieth National Conference on Artificial Intelligence (Landwehr et al., 2005).

schemes is that the resulting probabilistic logical or relational model is easier to understand and to learn.

In relational learning or inductive logic programming, one typically induces a set of rules (so-called clauses). The resulting rule-set then defines a disjunctive hypothesis, as an instance is classified as positive if it satisfies the conditions of one of the rules. On the other hand, a probabilistic model defines a joint probability distribution over a class variable and a set of “attributes” or “features”, and the type of model constrains the joint probability distributions that can be represented. A straightforward but powerful idea to integrate these two approaches is to interpret the clauses or rules as propositional “features” over which a joint probability distribution can be defined. Using naïve Bayes as the probabilistic model, this translates into the statement that “*clauses are independent*”. This idea is not really new. It has been pursued by Pompe and Kononenko (1995) and Davis et al. (2004). However, in these existing approaches for combining ILP and naïve Bayes, one learns the model in two *separate* steps. First, the features or clauses are generated (for example using an existing inductive logic programming system such as ILP-R (Pompe and Kononenko, 1995)) and then the probability estimates for the naïve Bayes are determined. This actually corresponds to a *static propositionalization* approach, where the propositionalized problem is learned using naïve Bayes.

We propose a *different* and *novel* approach, in which feature construction is tightly integrated with naïve Bayes. The advantage of such a *dynamic propositionalization* is that the criterion according to which the features are generated is that of naïve Bayes. Our intention in this paper is to investigate how dynamic propositionalization compares to static propositionalization approaches that use naïve Bayes only to post-process the rule set. More precisely, we will investigate:

(Q1) Is there a gain in predictive accuracy of a dynamic propositionalization approach over its ILP baseline?

(Q2) If so, is the gain of dynamic propositionalization over its baseline larger than the gain of static propositionalization approaches?

(Q3) Can performance be improved by using a more expressive probabilistic model, such as tree augmented naïve Bayes?

(Q4) Is a dynamic propositionalization based on a simple rule learner competitive with advanced ILP approaches?

(Q5) Relational naïve Bayes methods such as 1BC2 (Flach and Lachiche, 2004) essentially employ all clauses within a given language bias as features in a probabilistic model and thus perform static propositionalization. Does dynamic propositionalization employ fewer features and perform better than these approaches?

To answer Questions **Q1–Q5**, we present the NFOIL and TFOIL systems. The former is the first system reported in the literature that tightly integrates feature construction and naïve Bayes (De Raedt and Kersting, 2004; Landwehr et al., 2005). It guides the structure search by the probabilistic score of naïve Bayes. This contrasts with static propositionalization approaches, in which the criteria employed for feature generation and classification are different.

NFOIL essentially performs a covering search in which one feature (in the form of a clause) is learned after the other, until adding further features does not yield improvements. The search heuristic is based on class conditional likelihood and clauses are combined with naïve Bayes. TFOIL is an extension of NFOIL that employs a tree augmented naïve Bayes, that is, it relaxes the naïve Bayes assumption to allow additional probabilistic dependencies between clauses.

Recently, two other approaches that integrate the search for structural rules and a probabilistic model have been proposed. For STRUCTURAL LOGISTIC REGRESSION (Popescul et al., 2003), aggregated SQL queries are used as features in a logistic regression model. The authors propose an integrated search algorithm; however, the higher computational cost of training a logistic regression model allows for a limited search only. In the experimental evaluation, the implementation of Structural Logistic Regression only searches for clauses with up to two literals, where the first literal is fixed in advance. Furthermore, no experimental comparison to static propositionalization approaches is provided. In 2005, Davis et al. presented the SAYU system, which also learns a set of first-order rules that are combined in a probabilistic model. Clauses are selected according to the area under the precision-recall curve. The parameters of the probabilistic model are estimated to maximize the likelihood. Unlike naïve Bayes and NFOIL, SAYU does not handle multi-class problems (cf. the discussing of related work in Section 7).

The rest of the paper is organized as follows. Section 2 introduces some basic concepts from inductive logic programming and the simple first-order rule learning algorithm FOIL. In Section 3, we lift FOIL's problem setting to the probabilistic case and introduce a combined probabilistic/logical model for this setting. Section 4 shows how the NFOIL algorithm for learning such a model can be obtained by modifying certain components of the FOIL algorithm. Section 5 discusses how to extend the NFOIL algorithm to incorporate tree augmented naïve Bayes, resulting in the TFOIL system. In Section 6 we experimentally evaluate the proposed methods on benchmark data sets from four different domains and specifically investigate how it compares to static propositionalization approaches. Finally, we conclude and touch upon related work.

2. FOIL: First Order Inductive Learning

The problem that we tackle in this paper is a probabilistic formulation of the traditional inductive logic programming (ILP) problem. In ILP, the goal is to induce first-order clausal hypotheses from a relational description of a set of labeled examples and background knowledge. As an illustrating example, consider the task of inducing a theory that distinguishes between mutagenic and non-mutagenic chemical compounds based on the description of their structure (cf. the well-known mutagenicity problem (Srinivasan et al., 1996)):

Example 1 *Consider the following background theory B:*

$$\begin{array}{ll}
atom(189, d189\_1, c, 22, -0.11) & bond(189, d189\_1, d189\_2, 7) \\
atom(189, d189\_2, c, 22, -0.11) & bond(189, d189\_2, d189\_3, 7) \\
atom(189, d189\_3, c, 27, 0.02) & bond(189, d189\_3, d189\_4, 7) \\
\ldots & \ldots \\
atom(189, d189\_26, o, 40, -0.38) & bond(189, d189\_18, d189\_26, 2)
\end{array}$$

and the example $mutagenic(189)$. *A possible hypothesis for this domain is*

$$mutagenic(X) \leftarrow atom(X, A, o, 40, C), bond(X, B, A, 2)$$
$$mutagenic(X) \leftarrow atom(X, A, c, 22, C), atom(X, B, E, 22, 0.02), bond(X, A, B, 7)$$

In the example, $atom/5$ and $bond/4$ are *predicates* (of arity 5 and 4 respectively) that identify relations, numbers and lower-case strings like $189, c, d189\_1$ are *constants* that identify objects and upper-case letters like X, A, B, C are logical *variables*. *Logical atoms* are predicates together with their arguments, for example $bond(189, d189\_1, d189\_2, 7)$. *Definite clauses*, such as

$$mutagenic(X) \leftarrow atom(X, A, o, 40, C), bond(X, A, B, 2),$$

consist of a head $mutagenic(X)$ and a body $atom(X, A, o, 40, C), bond(X, A, B, 2)$. The logical atoms in the body are also called *literals*. *Facts* are definite clauses with an empty body, for example, $atom(189, d189\_1, c, 22, -0.11)$. A definite clause is called *ground* if it does not contain any variables. A *hypothesis* is a set of clauses. A hypothesis is said to *entail* an example given the background knowledge if the example is a logical consequence of the definite clauses in the hypothesis and background knowledge. In the example, $mutagenic(189)$ is entailed by the hypothesis given the background knowledge.

(Probabilistic) inductive logic programming problems can be formalized in a general way as follows (De Raedt and Kersting, 2004):

Given

- a background theory B, in the form of a set of definite clauses $h \leftarrow b_1, \cdots, b_n$;
- a set of examples E, in the form of ground facts, classified into classes C;
- a language of clauses $\mathcal{L}$, which specifies the clauses that are allowed in hypotheses;
- a $covers(e, H, B)$ function, which returns the classification $covers(e, H, B)$ of an example e, w.r.t. a hypothesis H, and the background theory B;
- a $score(E, H, B)$ function, which specifies the quality of the hypothesis H w.r.t. the data E and the background theory;

Find

$$\arg\max_{H \subset \mathcal{L}} score(E, H, B) .$$

Traditional approaches to inductive logic programming (Muggleton and De Raedt, 1994) tackle a concept-learning problem, in which there are typically two classes, and the goal is to find a complete and consistent concept-description. This can be formalized within our framework by making the following choices for *covers* and *score*:

- $covers(e, H, B) =$ positive if $B \cup H \models e$ (i.e., e is entailed by $B \cup H$);
  otherwise, $covers(e, H, B) =$ negative;
- $score(E, H, B) =$ training set accuracy (or a related measure).

Using this definition of coverage, the hypothesis given in Example 1 covers the example $mutagenic(189)$ together with the background knowledge. This setting is incorporated in many well-known inductive logic programming systems such as FOIL (Quinlan, 1990), GOLEM (Muggleton and Feng, 1990), PROGOL (Muggleton, 1995) and TILDE (Blockeel and De Raedt, 1997).

Algorithm 1 Generic FOIL algorithm.

```
Initialize $H := \emptyset$
repeat
  Initialize $c := p(X_1, \cdots, X_n) \leftarrow$
  repeat
    for all $c' \in \rho(c)$ do
      compute $score(E, H \cup \{c'\}, B)$
    end for
    let $c$ be the $c' \in \rho(c)$ with the best score
  until stopping criterion
  add $c$ to $H$
  $E := update(E, H)$
until stopping criterion
output $H$
```

FOIL, like many inductive logic programming systems, follows a *greedy* and *incremental* approach to induce a hypothesis. Such an algorithm is described in Algorithm 1. It repeatedly searches for clauses that score well with respect to the data set and the current hypothesis and adds them to the hypothesis. In the *update* function, the set of training examples can be updated after a clause has been learned. In the inner loop, the algorithm greedily searches for a clause that scores well. To this aim, it employs a general-to-specific hill-climbing search strategy. To generate the specializations of the current clause c, a so-called refinement operator ρ under θ-subsumption is employed. A clause c_1 θ-subsumes a clause c_2 if and only if there is a substitution θ such that $c_1\theta \subseteq c_2$. A substitution is a set $\{V_1/t_1, \ldots, V_l/t_l\}$ where the V_i are different variables and the t_i are terms, and the application of the substitution replaces the variables $V_1, \ldots, V_l$ by the corresponding terms $t_1, \ldots, t_l$. The most general clause is $p(X_1, \cdots, X_n) \leftarrow$ where p/n is the predicate being learned and the X_i are different variables. The refinement operator specializes the current clause $h \leftarrow b_1, \cdots, b_n$. This is typically realized by either adding a new literal l to the clause yielding $h \leftarrow b_1, \cdots, b_n, l$ or by applying a substitution θ yielding $h\theta \leftarrow b_1\theta, \cdots, b_n\theta$.

This type of algorithm has been successfully applied to a wide variety of problems in inductive logic programming. In classical FOIL, it is further simplified to a *separate-and-conquer* approach: examples that are covered by a learned clause are removed from the training data, and the score of a clause can be computed without respect to the current hypothesis; that is, $score(E, H \cup \{c'\}, B)$ simplifies to $score(E, c', B)$. Many different scoring functions and stopping criteria have been employed. The original FOIL algorithm uses information gain based on the number of positive and negative tuples bound to clauses as the scoring function (Quinlan, 1990), and a stopping criterion based on the minimum description length principle (Rissanen, 1978). MFOIL, a variant of FOIL that was particularly designed to cope with noise in the training data, uses the m-estimate for scoring clauses and a significance-based stopping criterion (Lavrač and Džeroski, 1994).

3. Integrating Naïve Bayes and FOIL: Problem Specification

Let us now discuss how to integrate the naïve Bayes method in FOIL's problem specification. This will be realized by modifying the *covers* and *score* functions in the inductive logic programming setting. All other elements, such as the examples, background theory and language of clauses $\mathcal{L}$ will—in principle—be untouched. However, the set of clauses defining a hypothesis is augmented with a set of parameters that quantify the probabilistic model. For easy of exposition, we will first discuss this for the case that the clauses are combined using naïve Bayes, that is, the nFOIL system. The extension to the tree-augmented naïve Bayes model (tFOIL) will be presented in Section 5.

3.1 A Probabilistic *covers* Function

In the setting of learning from probabilistic entailment (De Raedt and Kersting, 2004), the notion of coverage is replaced by a probability. We will use $\mathbf{P}$ to denote a probability distribution as in $\mathbf{P}(\mathbf{x})$, and P to denote a probability value as in $P(x)$, where x is a state of the random variable $\mathbf{x}$.

The probabilistic covers relation is then defined as the likelihood of the example, conditioned on the hypothesis and the background theory:

$$covers(e,H,B) = P(e \mid H,B), \tag{1}$$

where B is defined as before and $H = (H_C, H_\lambda)$ is an augmented hypothesis, consisting of a clause set H_C and an associated probabilistic model H_λ. An example e is of the form $p(X_1,\cdots,X_n)\theta = true$ or $p(X_1,\cdots,X_n)\theta = false$. Abusing notation—when the context is clear—we will sometimes refer to the example as θ, and say that the random variable $\mathbf{p}$ (class label) takes on the value $p\theta$ (true or false) for example θ.

We now still need to define $P(e \mid H,B)$. The key idea for realizing this is that we interpret the clauses in H together with the example e as queries or features. More formally, let H contain a set of clauses defining the predicate p. Then for each clause c of the form

$$p(X_1,\cdots,X_n) \leftarrow b_1,\cdots,b_n$$

we view the query $q_c = \leftarrow b_1,\cdots,b_n$ as a boolean feature or attribute. Applied to an example θ these queries become instantiated, $q_c\theta = \leftarrow b_1\theta,\cdots,b_n\theta$, and either succeed or fail in the background theory B. As for the class label, we will say that $q_c\theta$ is the observed (boolean) value of the random variable $\mathbf{q_c}$.

Example 2 *Assume that the background theory is given as in Example 1, and the query q_c under consideration is*

$$\leftarrow atom(X,A,o,40,C), bond(X,B,A,2)$$

For the example $\theta = \{X/189\}$, the instantiated clause

$$\leftarrow atom(189,A,o,40,C), bond(189,B,A,2)$$

succeeds in the background theory, so the boolean random variable $\mathbf{q_c}$ takes on value $q_c\theta = true$ for this example.

The probabilistic model H_λ of H specifies a distribution over the random variables $\mathbf{p}$ and $\mathbf{q_c}$. The observed (boolean) value of $\mathbf{q_c}$ is $q_c\theta$. We now define

$$\begin{aligned} P(e \mid H,B) &= P_\lambda(p\theta | q_1\theta,\cdots,q_k\theta) \\ &= \frac{P_\lambda(q_1\theta,\cdots,q_k\theta | p\theta)\cdot P_\lambda(p\theta)}{P_\lambda(q_1\theta,\cdots,q_k\theta)} \end{aligned}$$

where $H_C = \{q_1, ..., q_k\}$ and $\mathbf{P}_\lambda(\mathbf{p}, \mathbf{q_1}, ..., \mathbf{q_k})$ is the distribution defined by H_λ.

Now it becomes possible to state the naïve Bayes assumption

$$\mathbf{P}_\lambda(\mathbf{q_1}, ..., \mathbf{q_k} | \mathbf{p}) = \prod_i \mathbf{P}_\lambda(\mathbf{q_i} | \mathbf{p})$$

and apply it to our *covers* function (Equation 1):

$$P(e \mid H, B) = \frac{\prod_i P_\lambda(q_i\theta | p\theta) \cdot P_\lambda(p\theta)}{P_\lambda(q_1\theta, \cdots, q_k\theta)}.$$

At the same time, this equation specifies the parameters H_λ of the augmented model H, which are the distributions $\mathbf{P}_\lambda(\mathbf{q_i}|\mathbf{p})$ and the prior class distribution $\mathbf{P}_\lambda(\mathbf{p})$ (note that $\mathbf{P}_\lambda(\mathbf{q_1}, ..., \mathbf{q_k})$ can be computed by summing out).

Example 3 *Reconsider the mutagenicity example, and assume that the hypothesis is as sketched before. Then the queries q_1 and q_2 are*

$$mutagenic(X) \leftarrow atom(X, A, o, 40, C), bond(X, B, A, 2)$$
$$mutagenic(X) \leftarrow atom(X, A, c, 22, C), atom(X, B, E, 22, 0.02), bond(X, A, B, 7)$$

and the target predicate p is "mutagenic(X)". Now assume a naïve Bayes model with probability distributions $\mathbf{P}_\lambda(\mathbf{q_i}|\mathbf{p})$ given as

$$\begin{array}{lcl} P_\lambda(\mathbf{p} = t) = 0.6, & & \\ P_\lambda(\mathbf{q_1} = t | \mathbf{p} = t) = 0.7 & , & P_\lambda(\mathbf{q_1} = t | \mathbf{p} = f) = 0.4, \\ P_\lambda(\mathbf{q_2} = t | \mathbf{p} = t) = 0.2 & , & P_\lambda(\mathbf{q_2} = t | \mathbf{p} = f) = 0.1. \end{array}$$

Summing out yields

$$\begin{array}{ll} P_\lambda(\mathbf{q_1} = t, \mathbf{q_2} = t) = 0.10, & P_\lambda(\mathbf{q_1} = t, \mathbf{q_2} = f) = 0.48, \\ P_\lambda(\mathbf{q_1} = f, \mathbf{q_2} = t) = 0.06, & P_\lambda(\mathbf{q_1} = f, \mathbf{q_2} = f) = 0.36 \end{array}$$

where t (f) denotes true (false). For the positively labeled example $\theta = \{X/189\}$, q_1 succeeds and q_2 fails: $p\theta = true$, $q_1\theta = true$, $q_2\theta = false$. Thus,

$$P(e \mid H, B) = \frac{P_\lambda(q_1\theta | p\theta) \cdot P_\lambda(q_2\theta | p\theta) \cdot P_\lambda(p\theta)}{P_\lambda(q_1\theta, q_2\theta)} = \frac{0.7 \cdot 0.8 \cdot 0.6}{0.48} = 0.7.$$

3.2 The *score* Function

As scoring function, we employ the likelihood $P(E \mid H, B)$ of the data given the model and the background knowledge, and we assume also that the instances are independently and identically distributed (i.i.d.). So, we want to find the hypothesis H that maximizes

$$P(E \mid H, B) = \prod_{e \in E} P(e | H, B).$$

To evaluate a set of clauses H_C in the generic FOIL algorithm, the scoring function has to be changed accordingly, to

$$score(E, H_C, B) = P(E \mid H, B)$$

where H_C has been augmented with an optimal probabilistic model H_λ. This is discussed in more detail in Section 4.

3.3 Multi-Class Problems

Because we started from ILP, we have so far formulated the learning problem as a binary classification task. In ILP, multiple classes are usually encoded with a predicate $p(X,C)$ where C is an *output variable* that is instantiated with a class $c_i \in \{c_1,...,c_m\}$. One hypothesis is then induced per class using target predicate $p(X,c_i)$. However, combining these hypothesis to classify new instances is generally non-trivial, as *conflict resolution* strategies are needed to resolve conflicting predictions from the individual hypotheses. The same holds for approaches that use ILP to generate a rule set (Pompe and Kononenko, 1995; Davis et al., 2004) or an ILP technique such as bottom clauses to generate candidate rules for selection (Davis et al., 2005).

On the other hand, probabilistic models such as (tree augmented) naïve Bayes can deal with multi-class problems very naturally: the binary class variable is simply replaced by a multi-valued one. This directly carries over to the integrated model presented above. For a multi-class problem with classes $\{c_1,...,c_m\}$, the examples are of the form $p\theta = p(x,c_i)$ and the random variable $\mathbf{p}$ in the naïve Bayes model takes on some value $p\theta \in \{c_1,...,c_m\}$. No conflict resolution strategies are needed, as the naïve Bayes model directly returns a class $c_i \in \{c_1,\ldots,c_m\}$. Handling multi-class problems is part of the nFOIL and tFOIL implementations and we report on experimental results in Section 6.

4. Integrating Naïve Bayes and FOIL: Learning

The goal of learning is to identify a hypothesis H that maximizes the *score* function. More formally, the model space under consideration is

$$\mathcal{H} = \{(H_C,H_\lambda)|H_C \subseteq \mathcal{L}, H_\lambda \in \mathcal{M}_C\}$$

where $\mathcal{M}_C$ is the space of all naïve Bayes models over the clauses in H_C. The optimization problem is to find

$$\begin{aligned} H^* &= \arg\max_H P(E \mid H,B) \\ &= \arg\max_{H_C,H_\lambda} P(E \mid (H_C,H_\lambda),B), \end{aligned}$$

a hypothesis that jointly optimizes the clause set (structure) and probabilistic model (parameters such as the $\mathbf{P}_\lambda(\mathbf{q_i}|\mathbf{p})$ appearing in Example 3). Roughly speaking, the existing approaches pursued by Pompe and Kononenko (1995) and Davis et al. (2004) solve this task in a two-step process: First, H_C^* is found (using a standard ILP system, and thus some deterministic score) and fixed; second, the parameters for the fixed structure are optimized using a probabilistic score (usually, maximum likelihood):

```
for all $H_C \in candidates(\mathcal{L})$ do
    compute $ilp\text{-}score(E,H_C,B)$
end for
$H_C^* = \arg\max_{H_C} ilp\text{-}score(E,H_C,B)$
find $H_\lambda^* = \arg\max_{H_\lambda} score(E,(H_C^*,H_\lambda),B)$
```

As a consequence, it is unclear which score is being maximized at the *global* level.

A more principled way of solving such optimization problems relies on a nested approach: Evaluate structures H_C which are each augmented with optimal parameters, and select the best structure H_C^*. Thus, the task of structure selection involves augmenting a given structure with optimal parameters:

```
**for** all $H_C \in candidates(\mathcal{L})$ **do**
    find $H_\lambda^* = \arg\max_{H_\lambda} score(E,(H_C,H_\lambda),B)$
    $score(H_C) := score(E,(H_C,H_\lambda^*),B)$
**end for**
$H_C^* = \arg\max_{H_C} score(H_C)$
```

We will follow this more principled approach of guiding the search for the structure directly by the probabilistic objective function. The rest of this section shows how this can be realized by modifying the original search technique used in FOIL. This will first be presented for learning the basic nFOIL model. An extension of the learning algorithm that accounts for the tree augmented naïve Bayes structure in tFOIL will be discussed in Section 5.

4.1 Adapting FOIL

Like FOIL, nFOIL searches a set of clauses greedily, and a single clause in a general-to-specific manner using a refinement operator. The main difference in the search technique is that FOIL can use a *separate-and-conquer* approach. Because the final model in FOIL is the disjunction of the learned clauses (where every clause covers a certain subset of examples), it holds that

1. Examples that are already covered do not have to be considered when learning additional clauses: $update(E,H) = E \setminus covered(H)$.

2. (Non-recursive) clauses already learned do not need to be considered when scoring additional clauses: $score(E,H \cup \{c'\},B) = score(E,\{c'\},B)$.

Here, $score(E,\{c'\},B)$ is some accuracy-related measure such as information gain or the m-estimate. In nFOIL, such a separate-and-conquer approach is not possible because every clause can affect the likelihood of all examples. Consequently, the nFOIL algorithm can be obtained from FOIL by changing two components in the generic FOIL algorithm (see Algorithm 1):

1. The set of examples is not changed after learning a clause: $update(E,H) = E$.

2. An additional clause c' has to be scored together with the current model:

$$score(E,H_C \cup \{c'\},B) = P(E \mid H',B)$$

   where H' has clauses $H_C \cup \{c'\}$ and optimal parameters H'_λ.

We also have to modify the stopping criterion. The most basic stopping criterion for FOIL stops when all positive examples are covered. This is replaced in nFOIL by stopping if the change in score when adding a clause falls below a certain threshold. In general, this simple criterion might lead to overfitting. We therefore also investigated post-pruning the learned hypothesis (see Section 6). In the experiments, however, this basic stopping criterion worked surprisingly well.

4.2 Parameter Estimation: An Approximation

To evaluate a set of clauses $H_C = \{q_1,...,q_k\}$ by $score(E,H_C,B)$, one needs to solve the "inner" optimization problem of finding optimal parameters H_λ for the naïve Bayes model over the random variables $\{\mathbf{q_1},...,\mathbf{q_k}\}$:

$$\begin{aligned} H_\lambda^* &= \arg\max_{H_\lambda} P(E \mid H,B) \\ &= \arg\max_{\lambda} \prod_{\theta\in E} P_\lambda(p\theta|q_1\theta,\cdots,q_k\theta) \\ &= \arg\max_{\lambda} \prod_{\theta\in E} \frac{\prod_j P_\lambda(q_j\theta \mid p\theta)\cdot P_\lambda(p\theta)}{P_\lambda(q_1\theta,...,q_k\theta)}. \end{aligned} \tag{2}$$

However, these are the parameters maximizing the *conditional likelihood* of the observed class labels given the model. Usually, naïve Bayes (as a generative model) is trained to maximize the *likelihood*

$$\begin{aligned} \prod_{\theta\in E} P_\lambda(p\theta,q_1\theta,...,q_k\theta) &= \prod_{\theta\in E} P_\lambda(p\theta|q_1\theta,\cdots,q_k\theta)\cdot P_\lambda(q_1\theta,...,q_k\theta) \\ &= \prod_{\theta\in E}\prod_j P_\lambda(q_j\theta \mid p\theta)\cdot P_\lambda(p\theta). \end{aligned} \tag{3}$$

Maximum *likelihood* parameters can be computed easily by

$$P_\lambda(\mathbf{q_i}=q_i|\mathbf{p}=p) = \frac{n(\mathbf{q_i}=q_i,\mathbf{p}=p)}{n(\mathbf{p}=p)}$$

where $n(X)$ are the *counts*, that is, the number of examples for which the query X succeeds. However, there is no closed-form solution for maximum *conditional likelihood* parameters, and relatively slow iterative optimization algorithms have to be used.

Could we use the likelihood as defined by Equation (3) as the score? The problem is that this term is dominated by $P_\lambda(q_1\theta,...,q_k\theta)$, the likelihood of the propositional data set that is obtained by evaluating all features. This likelihood is easily maximized for very non-uniform distributions over the features values. In fact, it can be maximized to 1 if all features are constant (always succeed or always fail). Such features are of course completely uninformative with respect to predicting class labels. In contrast, in Equation (2) the likelihood is corrected by the term $P_\lambda(q_1\theta,...,q_k\theta)$, and in this way informative features are selected. Example 4 illustrates this situation for a single feature. Note that in the setting of parameter estimation from propositional data this problem does not occur as the distribution over the attribute values is determined by the (fixed) training data and cannot be changed by the learning algorithm. However, similar considerations apply to feature selection problems for (propositional) probabilistic models.

Example 4 *Consider the following queries q_1 and q_2:*

$$\begin{aligned} &\leftarrow true \\ &\leftarrow perfect\text{-}literal(X) \end{aligned}$$

Query q_1 succeeds on all examples. Assume that query q_2 succeeds on all positive and no negative examples, and that half of the examples are positive. Given maximum likelihood parameters, the

models H_1/H_2 consisting of only q_1/q_2 actually have the same *likelihood, while conditional likelihood correctly favors H_2:*
For any example θ*,*

$$\begin{aligned} P_\lambda(p\theta, q_1\theta) &= P_\lambda(q_1\theta|p\theta)\cdot P_\lambda(p\theta) = 1\cdot 0.5 = 0.5, \\ P_\lambda(p\theta, q_2\theta) &= P_\lambda(q_2\theta|p\theta)\cdot P_\lambda(p\theta) = 1\cdot 0.5 = 0.5 \end{aligned}$$

as $P_\lambda(q_1\theta|p\theta) = P_\lambda(q_2\theta|p\theta) = 1$*. On the other hand,*

$$\begin{aligned} P_\lambda(p\theta|q_1\theta) &= \frac{P_\lambda(q_1\theta|p\theta)\cdot P_\lambda(p\theta)}{P_\lambda(q_1\theta)} = \frac{1\cdot 0.5}{1} = 0.5, \\ P_\lambda(p\theta|q_2\theta) &= \frac{P_\lambda(q_2\theta|p\theta)\cdot P_\lambda(p\theta)}{P_\lambda(q_2\theta)} = \frac{1\cdot 0.5}{0.5} = 1 \end{aligned}$$

as $P_\lambda(q_1\theta) = 1$ *but* $P_\lambda(q_2\theta) = 0.5$.

Furthermore, a similar problem has been noted by Grossman and Domingos (2004) when learning the structure of Bayesian networks. Here, likelihood maximization leads to over-connected structures, a problem which is also solved by maximizing conditional likelihood. Because finding maximum conditional likelihood parameters is computationally too expensive, Grossman and Domingos propose a "mixed" approach: use conditional likelihood as score, but set parameters to their maximum likelihood values (seen as an approximation to maximum conditional likelihood values).

For nFOIL, we follow the same approach. Parameters are estimated to maximize the likelihood (Equation (3)), while the conditional likelihood, see Equation (2), is retained as score for selecting the features. Assuming that the naïve Bayes assumption is correct, the maximum likelihood estimates of the parameters will approach the maximum conditional likelihood estimates as the number of training examples goes to infinity. This is because the maximum likelihood estimate $P_\lambda(\mathbf{p}\theta, \mathbf{q_1}\theta, ..., \mathbf{q_k}\theta)$ approaches the true joint probability $P(\mathbf{p}\theta, \mathbf{q_1}\theta, ..., \mathbf{q_k}\theta)$ which determines the true conditional $P(\mathbf{p}\theta|\mathbf{q_1}\theta, ..., \mathbf{q_k}\theta)$ that also maximizes the conditional likelihood (Friedman and Goldszmidt, 1996).

This means that parameter estimates are easily obtained by counting, and computational costs for evaluating a hypothesis are dominated by computing for each query the set of examples on which it succeeds. Given this information, the computational cost of scoring a partial clause q_{k+1} in nFOIL is $O(C\cdot n)$ where C is the number of classes and n the number of examples. This assumes appropriate caching mechanisms on the parameters of $\{q_1, ..., q_k\}$ and on partial products of the scoring function (Equation (2)). Thus, evaluating a clause in nFOIL involves basically the same computational costs as scoring in FOIL. FOIL, however, profits from its separate-and-conquer approach; the number of examples is reduced after each iteration. Furthermore, the actual runtime depends on the path in the search space that is taken by the algorithm, which can be different for nFOIL because it may learn different clauses. Nevertheless, the computational complexity of nFOIL is roughly similar to that of FOIL.

5. tFOIL: Relaxing the Naïve Bayes Assumption

The naïve Bayes model employed in nFOIL corresponds to the strong assumption that the probability of an example satisfying one query is independent of its probability to satisfy another query,

Algorithm 2 TFOIL algorithm.

```
Initialize $H := \emptyset$
repeat
  Initialize $c := p(X_1, \cdots, X_n) \leftarrow$
  repeat
    for all $c' \in \rho(c)$ do
      for all $q \in H$ do
        compute $s(c', q) = score(E, H \cup \{c'\}, q \rightarrow c', B)$
      end for
      let $pa(c')$ be the $q \in H$ with maximum $s(c', q)$
    end for
    let $c$ be the $c' \in \rho(c)$ with maximum $s(c', pa(c'))$
  until stopping criterion
  add $c$ with dependency $pa(c') \rightarrow c'$ to $H$
until stopping criterion
output $H$
```

given the class of the example:

$$\mathbf{P}_\lambda(\mathbf{q_1}, ..., \mathbf{q_k} | \mathbf{p}) = \prod_i \mathbf{P}_\lambda(\mathbf{q_i} | \mathbf{p}). \tag{4}$$

Although it has been shown that naïve Bayes can perform well in practice even if this assumption is violated, better models can sometimes be constructed by relaxing this strict independence assumption (Friedman and Goldszmidt, 1996). Tree augmented naïve Bayes (TAN) models generalize naïve Bayes by allowing additional dependencies, but are still significantly more restrictive than full Bayesian networks. The restriction compared to a full Bayesian network is that the additional dependencies form a tree (i.e, every query node has at most one additional parent). This means that the number of parameters of a TAN model is in $O(\#nodes)$ as compared to $O(2^{\#nodes})$ for a full Bayesian network, and learning them is generally easier. In this section, we will discuss the TFOIL algorithm, which extends the NFOIL algorithm presented above to employ tree augmented naïve Bayes. Under the TAN assumption, Equation 4 is relaxed to

$$\mathbf{P}_\lambda(\mathbf{q_1}, ..., \mathbf{q_k} | \mathbf{p}) = \prod_i \mathbf{P}_\lambda(\mathbf{q_i} | \mathbf{p}, \mathbf{q_{pa(i)}})$$

where $\mathbf{q_{pa(i)}}$ is the additional parent of the node $\mathbf{q_i}$ in the TAN model. In analogy to Section 3, the *covers* function can be re-derived as

$$\begin{aligned} covers(E,H,B) =& P(e \mid H, B) \\ =& \frac{\prod_i P_\lambda(q_i\theta | p\theta, q_{pa(i)}\theta) \cdot P_\lambda(p\theta)}{P_\lambda(q_1\theta, \cdots, q_k\theta)}. \end{aligned}$$

Otherwise, the problem specification outlined in Section 3 directly carries over to TFOIL.

In contrast to NFOIL, learning a TFOIL model involves the additional task of selecting a TAN structure over the random variables representing the logical queries. In general, TFOIL follows the same integrated search strategy as outlined for NFOIL in Section 4. However, when evaluating a set of clauses $H_C = \{q_1,...,q_k\}$ in TFOIL, for every clause $q_i \in H_C$ we have to decide on a $q_j = q_{pa(i)} \in H_C$ for which a dependency $q_{pa(i)} \rightarrow q_i$ is added. Of course, this could be accomplished by running the standard TAN structure learning algorithm, which finds a maximum-likelihood structure in polynomial time (Friedman and Goldszmidt, 1996). However, the incremental way in which a theory H is learned and query nodes are added to the probabilistic model suggests a faster, incremental (though heuristic) way of learning the TAN structure. Rather than re-learning the TAN graph from scratch every time a new candidate clause q_i is scored, the subgraph over the existing hypothesis $H = \{q_1,...,q_{i-1}\}$ is kept fixed, and all existing clauses $q_j \in H$ are considered as possible parents for the clause q_i. Out of these candidate graph structures, the one maximizing $score(E,H,B)$ is chosen, that is, the maximum-likelihood extension of the existing graph on H.

This approach is outlined in Algorithm 2. The function $score(E,H \cup \{c'\},q \rightarrow c',B)$ returns the score of the TAN model over $H \cup \{c'\}$, where **q** is the additional parent of **c**$'$. For every candidate clause c', the best parent $pa(c')$ is identified and the c' with highest score is added to H with the dependency $pa(c') \rightarrow c'$. Comparing Algorithm 2 with the NFOIL algorithm which follows the template of Algorithm parent of a clause that is added to the model. The computational complexity of scoring the $(k+1)th$ clause q_{k+1} in TFOIL is $O(C \cdot k \cdot n)$, as all k existing clauses are considered as possible parents.

6. Experiments

In the following two subsections, we will describe the data sets and algorithms used to experimentally investigate the Questions **Q1**–**Q5** posed in the introduction (cf. Section 1). Section 6.3 then presents and discusses the results.

6.1 Data Sets

We conducted experiments on eight data sets from four domains. Three domains are binary classification tasks, and one is a multi-class problem. See Table 1 for an overview of the different data sets.

Mutagenesis (Srinivasan et al., 1996) is a well-known domain for structure-activity relation prediction. The problem is to classify compounds as mutagenic or not given their chemical structure described in terms of atoms, bonds, atom charge, and information about atom and bond types that have been generated by the molecular modeling package QUANTA. No additional numeric or handcrafted features of the compounds are used. The data set is divided into two sets: a regression friendly (**r.f.**) set with 188 entries (125 positives, 63 negatives) and a regression unfriendly (**r.u.**) set with 42 entries (13 positives and 29 negatives).

For **Alzheimer** (King et al., 1995), the aim is to compare 37 analogues of Tacrine, a drug against Alzheimer's disease, according to four desirable properties: inhibit **amine** re-uptake, low **toxicity**, high **acetyl** cholinesterase inhibition, and good **reversal** of scopolamine-induced memory deficiency. For any property, examples consist of pairs $pos(X,Y)/neg(X,Y)$ of two analogues indicating that X is better/worse than Y w.r.t. the property. The relation is transitive and anti-symmetric but not complete (for some pairs of compounds the result of the comparison could not be determined).

The **DSSTox** data set has been extracted from the EPA's DSSTox NCTRER Database (Fang et al., 2001). It contains structural information about a diverse set of 232 natural, synthetic and

Data Set	#Classes	#Examples	Majority Class	#Relations
Mutagenesis r.f.	2	188	66.5%	4
Mutagenesis r.u.	2	42	69.1%	4
Alzheimer amine	2	686	50.0%	20
Alzheimer toxic	2	886	50.0%	20
Alzheimer acetyl	2	1326	50.0%	20
Alzheimer memory	2	642	50.0%	20
DSSTox	2	232	56.5%	3
Diterpene	23	1530	23.5%	17

Table 1: Data sets used in experiments.

environmental estrogens and classifications with regard to their binding activity for the estrogen receptor. In our experiments, only structural information, that is, atom elements and bonds are used. Additionally, we provided a relation $linked(A_1, A_2, E, BT)$ in the background knowledge that represents that there is a bond of type BT from atom A_1 to atom A_2 and A_2 is of element E. This was done to reduce the lookahead problem for greedy search algorithms.

For **Diterpene** (Džeroski et al., 1998), the task is to identify the skeleton of diterpenoid compounds, given their C-NMR spectra that include the multiplicities and the frequencies of the skeleton atoms. Diterpenes are organic compounds of low molecular weight with a skeleton of 20 carbon atoms. They are of interest because of their use as lead compounds in the search for new pharmaceutical effectors. The data set contains information on 1530 diterpenes with known structure. There are in total 23 classes. We use the version where both relational and propositional information about the NMR spectra are available.

6.2 Algorithms and Methodology

We investigate the following learners:

- **nFOIL**
  An implementation of the nFOIL algorithm as outlined in Section 4.1. Instead of a greedy search a beam search with beam size $k = 5$ is performed. During the search for a clause, the algorithm also keeps a set C^* of the k best (partial) clauses found so far. If this set does not change from one refinement level to the next, the search is stopped and the best element $c \in C^*$ is returned. The search for additional clauses is stopped if the change in score between two successive iterations is less than 0.1%. A hypothesis is limited to contain at most 25 clauses and a clause to contain at most 10 literals. As most other ILP systems, nFOIL allows the specification of intensional background knowledge to be used in hypothesis. When classifying unseen examples, the class receiving the highest probability is returned (in particular, the default classification threshold of 0.5 is used in the binary case).

- **tFOIL**
  An implementation of the tFOIL algorithm as outlined in Section 5. The beam search and stopping criterion are implemented as for nFOIL.

- **mFOIL**
  mFOIL (Lavrač and Džeroski, 1994) is a variant of FOIL also employing beam search and different search heuristics. The beam size is set to $k = 5$, otherwise, default parameters are used. Note that mFOIL, unlike the other systems considered, by default allows negated literals in clauses.

- **ALEPH**
  ALEPH is an advanced ILP System developed by Ashvin Srinivasan.[1] It is based on the concept of *bottom clauses*, which are maximally specific clauses covering a certain example. The theory is then built from clauses that contain a subset of the literals found in a bottom clause. We used this standard mode of ALEPH for the binary domains **Mutagenesis**, **Alzheimer** and **DSSTox**. Additionally, ALEPH implements a tree learning algorithm, which we used for the multi-class domain **Diterpene**. The maximum number of literals in a clause was set to 10 instead of the default 4. Otherwise, default settings are used except on **DSSTox** as explained below.

- **1BC2**
  1BC2 is a naïve Bayes classifier for structured data (Flach and Lachiche, 2004). 1BC2 was run with a maximum number of 5 literals per clause, as larger values caused the system to crash. The decision threshold was optimized based on a five fold cross validation.

It would also be interesting to compare against the MACCENT system (Dehaspe, 1997), as maximum entropy models and naïve Bayes are somewhat related. Unfortunately, only an implementation of a propositional version of MACCENT is available, which only handles data in attribute-value (vector) format.[2] This version is not directly applicable to the relational data sets used in our study. We have therefore investigated a static propositionalization approach: frequent clauses were extracted from the relational data sets and then used as features in the propositional MACCENT system. More precisely, we have used a variant of the frequent pattern miner WARMR (Dehaspe et al., 1998), as WARMR patterns have shown to be effective propositionalization techniques on similar benchmarks in inductive logic programming (Srinivasan et al., 1999). The variant used was c-ARMR (De Raedt and Ramon, 2004), which allows to remove redundancies amongst the found patterns by focusing on so-called free patterns. c-ARMR was used to generate free frequent patterns in the data with a frequency of at least 20%. However, results obtained using this technique were on average not competitive with those of the other systems, and we decided not to include them.

To compare the different algorithms, we measure both accuracy and area under the ROC curve (see Fawcett, 2003), denoted as AUC. Accuracy is determined by a 10-fold cross-validation on all data sets except the small **Mutagenesis r.u.**, where a leave-one-out cross validation is used instead. ROC curves and AUC are determined from the cross-validation by pooling the rankings obtained on the different test folds. All algorithms were run on the same splits into training/test set for every fold. To test for significant differences in accuracy, a sampled paired t-test is applied to the results of the different folds.

Data Set	tFOIL	nFOIL	mFOIL	ALEPH	1BC2
Mutagenesis r.f.	79.7 ± 13.0	75.4 ± 12.3	76.6 ± 6.7	69.7 ± 11.9	**82.4**
Mutagenesis r.u.	83.3 ± 37.7	78.6 ± 41.5	71.4 ± 45.7	**85.7** ± 35.4	76.2
Alzheimer amine	**87.5** ± 4.4	86.3 ± 4.3	74.2 ± 5.9 ● ▲	70.9 ± 5.8 ● ▲	72.3
Alzheimer toxic	**92.1** ± 2.6	89.2 ± 3.4	81.9 ± 2.9 ● ▲	90.9 ± 1.4	83.4
Alzheimer acetyl	**82.8** ± 3.8	81.2 ± 5.2	75.2 ± 2.5 ● ▲	73.9 ± 3.4 ● ▲	73.4
Alzheimer memo.	**80.4** ± 5.3 △	72.9 ± 4.3 ●	60.9 ± 4.6 ● ▲	69.2 ± 5.3 ●	68.8
DSSTox	**78.5** ± 8.9	78.0 ± 9.1	70.4 ± 15.4	52.1 ± 11.2 ● ▲	64.7
Diterpene	**90.9** ± 2.1	90.8 ± 3.1	–	85.0 ± 3.6 ● ▲	81.9

Table 2: Cross-validated predictive accuracy results on all data sets. ●/○ indicates that tFOIL's mean is significantly higher/lower, and ▲/△ that nFOIL's mean is significantly higher/lower (paired sampled t-test, $p = 0.05$). Bold numbers indicate the best result on a data set. Note that the high variance on **Mutagenesis r.u.** is an artifact of the leave-one-out cross-validation. For **1BC2**, we do not test significance because the results on Mutagenesis and Diterpene are taken from Flach and Lachiche (2004).

6.3 Results

Table 2 shows the accuracy results for tFOIL, nFOIL, mFOIL, ALEPH and 1BC2 on all data sets. There is no result for mFOIL on Diterpene as it cannot handle multi-class problems. Comparing the results for nFOIL/tFOIL and mFOIL, the experiments clearly show a gain for dynamic propositionalization over the baseline ILP algorithm, giving an affirmative answer to Question **Q1**.

The comparison between nFOIL and its tree augmented extension tFOIL shows that—as in the propositional case—relaxing the naïve Bayes assumption can yield more accurate models in some cases. tFOIL always gains some predictive accuracy, with gains ranging from very slight (0.1 percentage points) to substantial (7.5 percentage points). Although only one of these gains is significant according to a paired sampled t-test on the folds, a simple sign test on the results of tFOIL and nFOIL (8/0) shows that tFOIL significantly outperforms nFOIL ($p < 0.01$). This affirmatively answers Question **Q3**: performance can be improved by using a more expressive model than naïve Bayes. Furthermore, tFOIL significantly outperforms ALEPH on five data sets, and both a sign test (7/1) and the average accuracy seem to favor tFOIL and nFOIL over ALEPH. This shows that relatively simple dynamic propositionalization approaches are competitive with more advanced ILP systems, giving an affirmative answer to Question **Q4**. The **DSSTox** domain seems to be particularly hard for the ALEPH system. Using standard settings, ALEPH only accepts rules if they cover no negative examples. In this case, for most folds it does not return any rules on the **DSSTox** data set, and only reaches 43.1% accuracy. The result reported above was obtained by setting the "noise" parameter to 10 (which means that a rule can cover up to 10 negative examples). The *minacc* parameter was left at its default value of 0.

To complement the study of predictive accuracy presented above, we investigate the performance of the probabilistic classifiers by means of ROC analysis. ROC curves evaluate how well the probability estimates produced by classifiers can discriminate between positive and negative

1. More information on ALEPH can be found at `http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/aleph_toc.html`
2. Luc Dehaspe, from personal communication.

Data Set	TFOIL	NFOIL	MFOIL	ALEPH	1BC2
Mutagenesis r.f.	**0.817**	0.809	0.791	0.713	0.816
Mutagenesis r.u.	0.753	0.737	0.645	**0.790**	0.729
Alzheimer amine	**0.945**	0.937	0.747	0.708	0.793
Alzheimer toxic	**0.983**	0.965	0.821	0.912	0.925
Alzheimer acetyl	**0.932**	0.916	0.759	0.752	0.815
Alzheimer memory	**0.913**	0.824	0.608	0.696	0.744
DSSTox	**0.789**	0.760	0.668	0.567	0.636

Table 3: Area under the ROC curve for all binary data sets. Bold numbers indicate the best result on a data set. Results for **1BC2** on Mutagenesis are taken from Flach and Lachiche (2004). Rankings are pooled over the different folds of a 10-fold cross-validation, except for Mutagenesis r.u. where a leave-one-out cross-validation is used instead.

examples (i.e., *rank* examples) without committing to a particular decision threshold. ROC curves provide more detailed information about performance than accuracy estimates, for example with regard to possible error trade-offs under variable misclassification costs (Provost et al., 1998). As ROC curves are only well-defined for binary classification problems, we do not report results for the multi-class data set **Diterpene**. For the ILP systems, ROC curves were produced by clause voting as introduced by Davis et al. (2004). In clause voting, the threshold that is varied is the number of clauses that have to cover an example before it is classified as positive (the default threshold being one).

AUC results, shown in Table 3, generally confirm the results obtained by accuracy analysis: TFOIL outperforms NFOIL across the board, and TFOIL/NFOIL generally produce better rankings than the ILP methods and 1BC2. Note that in some cases dynamic propositionalization achieves higher AUC scores than ILP systems even though it achieves lower accuracy (e.g., comparing NFOIL and ALEPH on the **Alzheimer toxic** data set). To investigate this behavior in more detail, ROC curves for the different methods on all binary data sets are shown. Figure 1 shows curves on the relatively large **Alzheimer amine**, **Alzheimer toxic**, **Alzheimer acetyl** and **Alzheimer memory** data sets. Here, ROC curves clearly fall into two groups: for the ILP systems MFOIL and ALEPH, the curve has one sharp angle and is otherwise mostly linear, while the curves for TFOIL, NFOIL and 1BC2 are convex almost across the whole range of the threshold parameter. This means that for the ILP systems there is one (namely, the default) decision threshold which offers a good trade-off between true positives and false positives, but ranking below and above this point is relatively poor. At the level of induced clauses, it indicates that clauses induced by the ILP systems are specific in the sense that positive examples are typically covered by only one clause—if the decision threshold in clause voting is set to more than one, the true positive rate drops rapidly. In contrast, ranking performance for the dynamic propositionalization systems and 1BC2 is more stable, meaning that these systems also offer good classification performance under varying misclassification costs (or, equivalently, class skew). This indicates that dynamic propositionalization approaches can make use of more diverse rule sets, which are helpful in ranking examples by providing additional information but would produce too many false-positive classifications if interpreted as a disjunctive hypothesis. We will provide further evidence for this claim below.

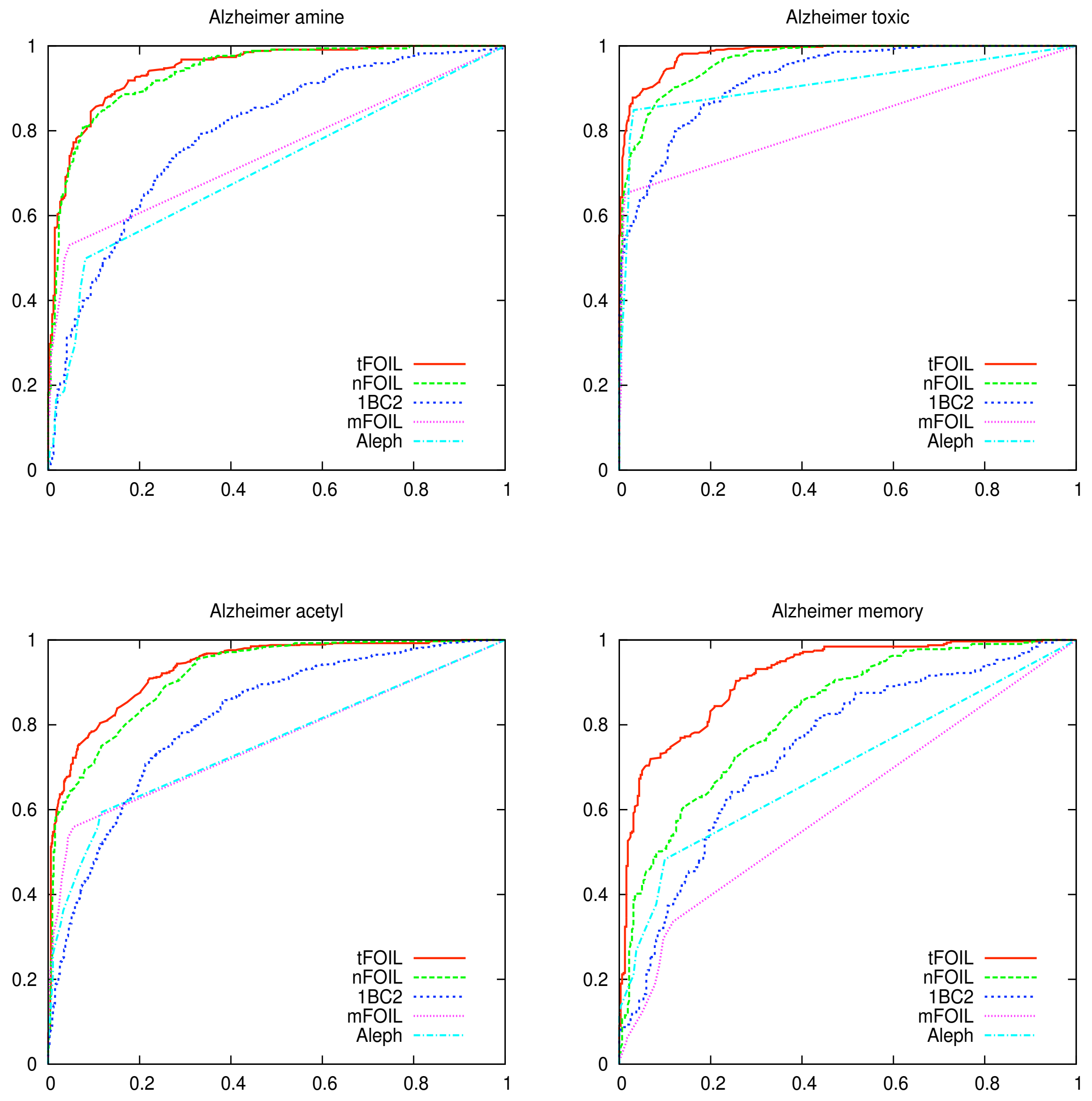


Figure 1: ROC curves on the Alzheimer amine, Alzheimer toxic, Alzheimer acetyl and Alzheimer memory data sets for TFOIL, NFOIL, MFOIL, ALEPH and 1BC2. Rankings are pooled over the different folds of a 10-fold cross-validation.

Figure 2 shows ROC curves on the three smaller data sets included in our study. On **Mutagenesis r.f.** and **DSSTox**, similar observations hold as noted above, although class separation is generally poorer and curves behave less well. On the very small **Mutagenesis r.u.** data set ranking performance is poor for all methods.

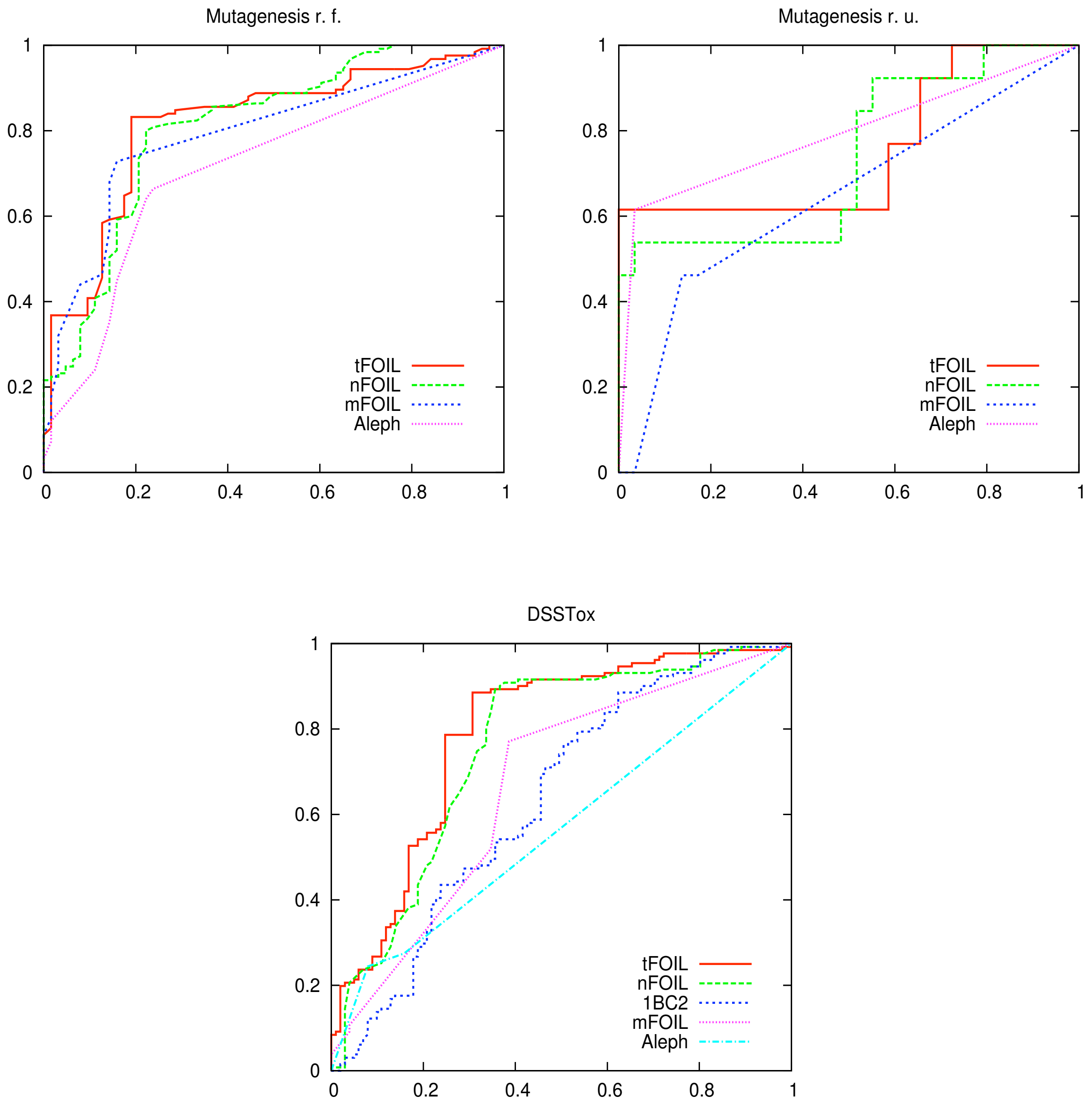


Figure 2: ROC curves on the Mutagenesis regression friendly, Mutagenesis regression unfriendly and DSSTox data sets for TFOIL, NFOIL, MFOIL, ALEPH and 1BC2. There is no curve for 1BC2 on the Mutagenesis data sets because results are taken from Flach and Lachiche (2004). Rankings are pooled over the different folds of a 10-fold cross-validation, except for Mutagenesis r.u. where a leave-one-out cross-validation is used instead.

	MFOIL		ALEPH		
Data Set	**+NB**	**+TAN**	**+NB**	**+TAN**	**NFOIL/disjunctive**
Mutagenesis r.f.	±0.0 ●	±0.0 ●	±0.0	±0.0	−8.9 ● ▲
Mutagenesis r.u.	±0.0	+2.4	±0.0	±0.0	−47.6 ● ▲
Alzheimer amine	−0.5 ● ▲	±0.0 ● ▲	±0.0 ● ▲	±0.0 ● ▲	−36.3 ● ▲
Alzheimer toxic	±0.0 ● ▲	+0.2 ● ▲	±0.0	±0.0	−39.2 ● ▲
Alzheimer acetyl	−0.4 ● ▲	±0.0 ● ▲	±0.0 ● ▲	+0.1 ● ▲	−31.2 ● ▲
Alzheimer memory	±0.0 ● ▲	±0.0 ● ▲	±0.0 ●	±0.0 ●	−22.9 ● ▲
DSSTox	+2.1	+1.7	+4.3 ● ▲	+4.3 ● ▲	−21.5 ● ▲
Diterpene	–	–	±0.0 ● ▲	−0.3 ● ▲	–

Table 4: Gain/loss in cross-validated predictive accuracy for the two-step methods MFOIL+NB/TAN, ALEPH+NB/TAN and NFOIL/DISJUNCTIVE over their corresponding baselines MFOIL, ALEPH and NFOIL. ●/○ indicates that TFOIL's mean accuracy is significantly higher/lower, and ▲/△ that NFOIL's mean accuracy is significantly higher/lower (paired sampled t-test, $p = 0.05$).

To investigate Question **Q2**, that is, to compare dynamic and static propositionalization approaches, two additional experiments were performed:

1. Learning a naïve Bayes or tree augmented naïve Bayes model over a set of clauses using a two-step approach: First, a set of rules is learned using MFOIL or ALEPH, and afterwards a (tree augmented) naïve Bayes model is built using these rules. This is a static propositionalization approach, where the propositionalized data is used as input for the probabilistic learner. Question **Q2** is whether these rules are less useful when combined with (tree augmented) naïve Bayes than the rules constructed by a dynamic propositionalization approach (NFOIL/TFOIL). For the training of the TAN model we used the local score based TAN implementation in WEKA 3.4.6 (Witten and Frank, 2000), which implements the learning algorithm of Friedman and Goldszmidt (1996).

2. Using the rules learned by NFOIL as a disjunctive hypothesis Q: For this, every rule q learned by NFOIL is evaluated on the training set. If it covers more positive than negative examples, q is added to Q, otherwise $not(q)$. The rule set Q is then evaluated as a disjunctive hypothesis on the test data. This technique can only be used for binary classification problems.

Table 4 shows the result of these experiments. It displays the average gain/loss of the two-step methods MFOIL+NB, ALEPH+NB and NFOIL/disjunctive over their corresponding baselines MFOIL, ALEPH and NFOIL. Significantly higher/lower mean accuracies of a method as compared to TFOIL/NFOIL are also indicated. On most data sets, applying naïve Bayes as a method for post-processing the learned rule set of MFOIL or ALEPH does not yield any improvement, with the exception of small gains on the **DSSTox** and **Mutagenesis r.u.** data sets for MFOIL. ALEPH+NB/TAN only improves on the original result of ALEPH on **DSSTox** by always predicting the majority class. Furthermore, NFOIL/TFOIL significantly outperform MFOIL+NB and MFOIL+TAN on the same data sets on which they significantly outperform MFOIL. In ROC space, post-processing rules with (tree augmented) Naïve Bayes can increase or decrease the performance

	MFOIL		ALEPH	
Data Set	**+NB**	**+TAN**	**+NB**	**+TAN**
Mutagenesis r.f.	+0.045	+0.036	+0.011	−0.034
Mutagenesis r.u.	−0.183	−0.183	−0.175	−0.063
Alzheimer amine	+0.001	+0.007	−0.013	−0.012
Alzheimer toxic	+0.003	+0.011	+0.006	+0.005
Alzheimer acetyl	+0.002	−0.003	−0.006	−0.010
Alzheimer memory	−0.010	−0.016	−0.010	−0.027
DSSTox	+0.054	+0.048	−0.095	−0.099

Table 5: Gain/loss in area under ROC curve for the two-step methods MFOIL+NB/TAN and ALEPH+NB/TAN over their corresponding baselines MFOIL and ALEPH. Rankings are pooled over the different folds of a 10-fold cross-validation, except for Mutagenesis r.u. where a leave-one-out cross-validation is used instead.

depending on the data set (Table 5). However, AUC scores of the static propositionalization approaches are on average much lower than those of dynamic propositionalization approaches.

Using the (possibly negated) rules learned by NFOIL as a disjunctive hypothesis strongly degrades performance. This is because NFOIL can make use of rules which have a very low accuracy individually but still help as additional features in the naïve Bayes. If these rules are used for disjunctive classification, they produce many false-positive classifications. In fact, on some domains (e.g, **Alzheimer**) all examples in the test set are always classified as positive. This shows that NFOIL uses significantly different rule sets to ILP approaches. Thus, we can answer **Q2** as follows:

> A dynamic propositionalization approach that selects rules based on the criterion of the probabilistic model performs better than static propositionalization approaches that post-process a rule set using a probabilistic learner. This is because dynamic propositionalization can make use of different/additional rules that are not considered by traditional ILP systems as they would produce too many false-positive classifications.

It remains to answer Question **Q5**, that is, to compare dynamic propositionalization approaches to relational naïve Bayes methods such as 1BC2 in terms of accuracy and the number of features they employ. With respect to predictive accuracy and AUC score, sign tests prefer TFOIL over 1BC2 (at 7/1 and 8/0, respectively). Theory complexity is hard to compare because of the different representations. For TFOIL/NFOIL and 1BC2, one complexity measure is the number of probability values attached to clauses. There are *#classes* probability values attached to each clause in NFOILand 1BC2, and $2 \cdot \#classes$ in TFOIL (*#classes* denotes the number of classes). Additionally, we have to specify the prior distribution over the class variable. The overall number of probability values to be specified is thus of the order of $O(\#clauses)$, and it is sufficient to compare the number of clauses. In the experiments, 1BC2 uses an order of magnitude more clauses than NFOIL. More precisely, 1BC2 uses more than 400 (in some cases even more than 1000) clauses whereas NFOIL is limited to using 25 clauses. This clearly shows that **Q5** can be answered affirmatively as well.

We furthermore investigated whether the proposed dynamic propositionalization approach sometimes constructs too many clauses and overfits the training data, as the stopping criterion is based on the training set score. We therefore tried post-pruning a learned hypothesis. Post-pruning is more

Algorithm 3 Post-pruning a hypothesis learned by NFOIL.

```
Initialize $H := \text{NFOIL}(E,B)$
repeat
  for all $c \in H$ do
    $s(c) :=$ cross-validate-accuracy$(H \setminus \{c\}, E, B)$
  end for
  $c^* := \arg\max_{c \in H} s(c)$
  $H := H \setminus \{c^*\}$
until cross-validated accuracy decreases
output $H$
```

	NFOIL		**NFOIL/pruning**	
Data Set	Accuracy	#clauses	Accuracy	#clauses
Mutagenesis r.f.	**75.4** ± 12.3	25.0	73.9 ± 12.1	17.7
Mutagenesis r.u.	78.6 ± 41.5	23.1	**85.7** ± 35.4	1.4
Alzheimer amine	**86.3** ± 4.3	24.7	85.0 ± 4.5	20.1
Alzheimer toxic	**89.2** ± 3.4	22.4	87.8 ± 3.6	16.6
Alzheimer acetyl	**81.2** ± 5.2	25.0	80.8 ± 4.2	20.5
Alzheimer memory	72.9 ± 4.3	24.9.5	**74.5** ± 4.3	20.4
DSSTox	78.0 ± 9.1	15.4	**79.3** ± 9.7	4.5
Diterpene	**90.8** ± 3.1	25.0	90.7 ± 3.1	23.4

Table 6: Cross-validated predicative accuracy results and average number of clauses in the final model for NFOIL and NFOIL/pruning. Bold numbers indicate the best result on a data set. The are no significant differences in mean accuracy between the two methods (paired sampled t-test, $p = 0.05$).

easily realized for NFOIL than for TFOIL, as the additional TAN structure in the TFOIL model prevents removal of rules which are parents of other rules. Rule post-pruning was carried out using the greedy algorithm outlined in Algorithm 3. The procedure *cross-validate-accuracy(H,E,B)* cross-validates the naïve Bayes model on the training data for a fixed set H of clauses and returns an accuracy estimate. The algorithm greedily drops clauses from H as long as this does not decrease the cross-validated accuracy estimate. Table 6 lists the accuracies of NFOIL and NFOIL/pruning which incorporates this rule post-pruning algorithm. There is some gain in accuracy on the small **Mutagenesis r.u.** and the **DSSTox** domain, although no differences in accuracy are significant at the $p = 0.05$ level. On these two data sets the number of features is also greatly reduced, while few or no features are pruned for the other data sets. To summarize, a clear overfitting behavior can not be observed except possibly on the very small **Mutagenesis r.u.** data set.

We conclude that our experimental study affirmatively answers Questions **Q1**–**Q5** posed in the introduction. The dynamic propositionalization approaches NFOIL and TFOIL yield more accurate models than simple ILP rule learning and static propositionalization approaches, and also compare favorably to the first order naïve Bayes system 1BC2 and one of the most advanced ILP systems, namely Aleph.

7. Related Work

The approaches that combine statistical learning with inductive logic programming techniques for addressing classification can be divided into three categories.

A first class of techniques are static propositionalization approaches. They start by generating a set of first order features and then use these features as attributes in a probabilistic model. Probabilistic models of different expressivity have been used, ranging from naïve Bayes (Pompe and Kononenko, 1995; Flach and Lachiche, 2004), to tree augmented naïve Bayes or full Bayesian networks as in Davis et al. (2004). The set of features is obtained either by taking all features within a pre-defined language bias, as in the 1BC system (Flach and Lachiche, 2004), or by running a traditional ILP algorithm (Pompe and Kononenko, 1995; Davis et al., 2004). Furthermore, aggregation-based feature construction methods such as RELAGGS (Krogel and Wrobel, 2001) and ACORA (Perlich and Provost, 2006) that search a relational feature space using aggregation operators fall into this group. In this class of techniques the feature construction and the statistical learning steps are performed consecutively and independent of one another, whereas in NFOIL and TFOIL they are tightly integrated. However, an initial step beyond static propositionalization has been taken in the work by Pompe and Kononenko (1997), where rules generated by an ILP system are post-processed by splitting and merging clauses in order to find a rule set that satisfies the naïve Bayes assumption.

A second class of techniques employs a rich probabilistic model such as a Probabilistic Relational Model (Getoor et al., 2001) or a higher-order probabilistic logic (Flach and Lachiche, 2004). The logical component (and hence the features) of such a model are fixed, and only the parameters are learned using statistical learning techniques. The work by Taskar et al. (2001) on using Probabilistic Relational Models for clustering and classification of relational data, the Relational Bayesian Classifier (Neville et al., 2003) and the 1BC2 system (Flach and Lachiche, 2004) fall into this category. The difference compared to NFOIL and TFOIL is that this class of techniques does not address structure learning or feature generation.

A third class of techniques (Popescul et al., 2003; Dehaspe, 1997) indeed tightly integrates the inductive logic programming step with the statistical learning step in a dynamic propositionalization approach. However, whereas the dynamic propositionalization methods presented in this paper employ the simplest possible statistical model, namely naïve Bayes, those approaches use more advanced (and hence computationally more expensive) statistical models such as logistic regression and maximum entropy modeling, which does seem to limit the application potential. For instance, Popescul et al. (2003) report that—in their experiments—they had to employ a depth limit of 2 when searching for features. The work on NFOIL and TFOIL is similar in spirit to these two approaches but is much more simple and therefore, we believe, also more appealing for the traditional classification task considered in inductive logic programming.

Probably the most closely related approach to the methods presented in this paper is the SAYU system (Davis et al., 2005). SAYU uses a "wrapper" approach where (partial) clauses generated by the refinement search of an ILP system are proposed as features to a (tree augmented) naïve Bayes, and incorporated if they improve performance. This means that feature learning and naïve Bayes are tightly coupled as in our approach. However, in SAYU the scores for feature and parameter selection are different, and the feature selection is based on a separate tuning set. The probabilistic model is trained to maximize the likelihood on the training data, while clause selection is based on the area under the precision-recall curve of the model on a separate tuning set. This contrasts with the approach of selecting features and parameters that jointly optimize a probabilistic score on the training data used in NFOIL and TFOIL. Davis et al. (2005) also report that a tree-augmented naïve Bayes model in SAYU does not significantly outperform a naïve Bayes, which contrasts with the results obtained in this study. Furthermore, SAYU cannot handle multi-class problems, while NFOIL and TFOIL handle them as naturally as naïve Bayes does.

Finally, there is also the approach of Craven and Slattery (2001) who combine several naïve Bayes models with FOIL. The decisions of naïve Bayes models are viewed as truth values of literals occurring in clauses. This work can be regarded as the inverse of the approach presented in this paper in that NFOIL/TFOIL employ naïve Bayes on top of logic, whereas Craven and Slattery employ naïve Bayes as a predicate in the logical definitions.

8. Conclusions

We have introduced the NFOIL and TFOIL systems, two dynamic propositionalization approaches that combine the simplest techniques from ILP and probabilistic learning. In an experimental study on several benchmark data sets, the proposed approaches were compared to the ILP systems MFOIL and ALEPH, static propositionalization approaches and the first-order naïve Bayes system 1BC2. Experimental results show that dynamic propositionalization is superior to static propositionalization and simple rule learning. Despite their simplicity, the proposed approaches are also competitive with the more advanced ILP system ALEPH. Moreover, our experiments indicate that the superior performance of dynamic propositionalization approaches is due to the fact that they can make use of more diverse rule sets than ILP systems. NFOIL and TFOIL are also particularly strong at ranking examples (as measured by ROC analysis), a task in which standard ILP systems performed rather poorly in our experiments.

Further exploring dynamic propositionalization as a way of combining (possibly more powerful) statistical learners with ILP search techniques is an interesting direction for future work.

Acknowledgments

We would like to thank the three anonymous reviewers for their helpful comments and suggestions, which have greatly improved this paper. We furthermore thank Romaric Gaudel for fruitful discussions and help with performing the experimental study. We acknowledge support for this work from the European Union under contract number FP6-508861, Applications of Probabilistic Inductive Logic Programming II.

References

Hendrik Blockeel and Luc De Raedt. Lookahead and discretization in ILP. In N. Lavrač and S. Džeroski, editors, *Proceedings of the Seventh International Workshop on Inductive Logic Programming (ILP-1997)*, volume 1297 of *Lecture Notes in Computer Science*, pages 77–84. Springer, 1997.

Mark Craven and Seán Slattery. Relational learning with statistical predicate invention: Better models for hypertext. *Machine Learning*, 43(1–2):97–119, 2001.

Jesse Davis, Irene M. Ong Vítor Santos Costa, David Page, and Inês Dutra. Using Bayesian classifiers to combine rules. In *Working Notes of the Third Workshop on Multi-Relational Data Mining (MRDM-2004) in conjunction with the Tenth ACM SIGKDD International Conference on Knowlege Discovery and Data Mining (KDD-2004)*, Seattle, Washington, USA, 2004.

Jesse Davis, Elizabeth Burnside, Inês de Castro Dutra, David Page, and Vítor Santos Costa. An integrated approach to learning Bayesian networks of rules. In Joao Gama, Rui Camacho, Pavel Brazdil, Alípio Jorge, and Luís Torgo, editors, *Proceedings of the Sixteenth European Conference on Machine Learning (ECML-2005)*, volume 3720 of *Lecture Notes in Computer Science*, pages 84–95. Springer, 2005.

Luc De Raedt and Kristian Kersting. Probabilistic logic learning. *ACM-SIGKDD Explorations*, 5 (1):31–48, 2003.

Luc De Raedt and Kristian Kersting. Probabilistic inductive logic programming. In S. Ben-David, J. Case, and A. Maruoka, editors, *Proceedings of the Fifteenth International Conference on Algorithmic Learning Theory (ALT-2004)*, volume 3244 of *Lecture Notes in Computer Science*, pages 19–36. Springer, 2004.

Luc De Raedt and Jan Ramon. Condensed representations for inductive logic programming. In *Proceedings of the Ninth International Conference on Principles of Knowledge Representation and Reasoning (KR-2004)*, Whistler, Canada, 2004. AAAI Press.

Luc Dehaspe. Maximum entropy modeling with clausal constraints. In N. Lavrač and S. Džeroski, editors, *Proceedings of the Seventh International Workshop on Inductive Logic Programming (ILP-1997)*, volume 1297 of *Lecture Notes in Computer Science*, pages 109–124. Springer, 1997.

Luc Dehaspe, Hannu Toivonen, and Ross D. King. Finding frequent substructures in chemical compounds. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-1998)*, New York City, New York, USA, 1998. AAAI Press.

Saso Džeroski, Steffen Schulze-Kremer, Karsten Heidtke, Karsten Siems, Dietrich Wettschereck, and Hendrik Blockeel. Diterpene structure elucidation from^{13}c NMR spectra with inductive logic programming. *Applied Artificial Intelligence, Special Issue on First-Order Knowledge Discovery in Databases*, 12:363–383, 1998.

Hong Fang, Weida Tong, Leming M. Shi, Robert Blair, Roger Perkins, William Branham, Bruce S. Hass, Qian Xie, Stacy L. Dial, Carrie L. Moland, and Daniel M. Sheehan. Structure-activity relationships for a large diverse set of natural, synthetic, and environmental estrogens. *Chemical Research in Toxicology*, 14(3):280–294, 2001.

Tom Fawcett. ROC graphs: Notes and practical considerations for data mining researchers, 2003.

Peter Flach and Nicholas Lachiche. Naive Bayesian classification of structured data. *Machine Learning*, 57(3):233–269, 2004.

Nir Friedman and Moises Goldszmidt. Building classifiers using Bayesian networks. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-1996), Vol. 2*, pages 1277–1284, Portland, Oregon, USA, 1996. AAAI Press / The MIT Press.

Lise Getoor and Ben Taskar, editors. *Statistical Relational Learning*. MIT Press, 2007. In press.

Lise Getoor, Nir Friedman, Daphne Koller, and Avi Pfeffer. Learning probabilistic relational models. In *Relational Data Mining*. Springer, 2001.

Daniel Grossman and Peter Domingos. Learning Bayesian network classifiers by maximizing conditional likelihood. In *Proceedings of the Twenty-First International Conference on Machine Learning (ICML-2004)*, pages 361–368, Banff, Canada, 2004. ACM Press.

Ross D. King, Ashvin Srinivasan, and Michael J, E Sternberg. Relating chemical activity to structure: An examination of ILP successes. *New Generation Computing*, 13(2,4):411–433, 1995.

Mark A. Krogel and Stefan Wrobel. Transformation-based learning using multirelational aggregation. In Céline Rouveirol and Michèle Sebag, editors, *Proceedings of the Eleventh International Conference on Inductive Logic Programming (ILP-2001)*, volume 2157 of *Lecture Notes in Computer Science*. Springer, 2001.

Niels Landwehr, Kristian Kersting, and Luc De Raedt. nFOIL: Integrating naïve Bayes and FOIL. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-2005)*, pages 795–800, Pittsburgh, Pennsylvania, USA, 2005. AAAI Press.

Nada Lavrač and Saso Džeroski. *Inductive Logic Programming*. Ellis Horwood, 1994.

Stephen Muggleton. Inverse entailment and Progol. *New Generation Computing, Special Issue on Inductive Logic Programming*, 13:245–286, 1995.

Stephen Muggleton. Stochastic logic programs. In *Advances in Inductive Logic Programming*. IOS Press, 1996.

Stephen Muggleton and Luc De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19/20:629–679, 1994.

Stephen Muggleton and Cao Feng. Efficient induction of logic programs. In *Proceedings of the First Conference on Algorithmic Learning Theory (ALT-1990)*, pages 368–381, Tokyo, Japan, 1990. Springer.

Jennifer Neville, David Jensen, and Brian Gallagher. Simple estimators for relational Bayesian classifiers. In *Proceedings of the Third IEEE International Conference on Data Mining (ICDM-2003)*, pages 609–612, Melbourne, Florida, USA, 2003. IEEE Computer Society.

Claudia Perlich and Foster Provost. Distribution-based aggregation for relational learning with identifier attributes. *Machine Learning*, 62:65–105, 2006.

Uros Pompe and Igor Kononenko. Naive Bayesian classifier within ILP-R. In *Proceedings of the Fifth International Workshop on Inductive Logic Programming (ILP-1995)*, pages 417–436, Tokyo, Japan, 1995.

Uros Pompe and Igor Kononenko. Probabilistic first-order classification. In N. Lavrač and S. Džeroski, editors, *Proceedings of the Seventh International Workshop on Inductive Logic Programming (ILP-1997)*, volume 1297 of *Lecture Notes in Computer Science*, pages 235–242. Springer, 1997.

Alexandrin Popescul, Lyle H. Ungar, Steve Lawrence, and David M. Pennock. Statistical relational learning for document mining. In *Proceedings of the Third IEEE International Conference on Data Mining (ICDM-2003)*, pages 275–282, Melbourne, Florida, USA, 2003. IEEE Computer Society.

Foster J. Provost, Tom Fawcett, and Ron Kohavi. The case against accuracy estimation for comparing induction algorithms. In *Proceeding of the Fifteenth International Conference on Machine Learning (ICML-1998)*, Madison, Wisconsin, USA, 1998. Morgan Kaufmann.

J. Ross Quinlan. Learning logical definitions from relations. *Machine Learning*, pages 239–266, 1990.

Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

Ashvin Srinivasan, Stephen Muggleton, Ross D. King, and Michael J. E. Sternberg. Theories for mutagenicity: A study of first-order and feature based induction. *Artificial Intelligence*, 85: 277–299, 1996.

Ashwin Srinivasan, Ross D. King, and Douglas W. Bristol. An assessment of ILP-assisted models for toxicology and the PTE-3 experiment. In Saso Dzeroski and Peter A. Flach, editors, *Proceedings of the Ninth Internatinal Workshop on Inductive Logic Programming (ILP-1999)*, volume 1634 of *Lecture Notes in Computer Science*. Springer, 1999.

Ben Taskar, Eran Segal, and Daphne Koller. Probabilistic clustering in relational data. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001)*, pages 870–878, Seattle, Washington, USA, 2001. Morgan Kaufmann.

Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implemenations*. Morgan Kaufmann, 2000.

Journal of Machine Learning Research 8 (2007) 509-547 Submitted 2/05; Revised 3/06; Published 3/07

A Stochastic Algorithm for Feature Selection in Pattern Recognition

Sébastien Gadat GADAT@MATHS.UPS-TLSE.FR
CMLA, ENS Cachan
61 avenue du président Wilson
CACHAN 94235 Cachan Cedex, FRANCE

Laurent Younes LAURENT.YOUNES@CIS.JHU.EDU
Center for Imaging Science
The Johns Hopkins University
3400 N-Charles Street
Baltimore MD 21218-2686, USA

Editor: Isabelle Guyon

Abstract

We introduce a new model addressing feature selection from a large dictionary of variables that can be computed from a signal or an image. Features are extracted according to an efficiency criterion, on the basis of specified classification or recognition tasks. This is done by estimating a probability distribution $\mathbb{P}$ on the complete dictionary, which distributes its mass over the more efficient, or informative, components. We implement a stochastic gradient descent algorithm, using the probability as a state variable and optimizing a multi-task goodness of fit criterion for classifiers based on variable randomly chosen according to $\mathbb{P}$. We then generate classifiers from the optimal distribution of weights learned on the training set. The method is first tested on several pattern recognition problems including face detection, handwritten digit recognition, spam classification and micro-array analysis. We then compare our approach with other step-wise algorithms like random forests or recursive feature elimination.

Keywords: stochastic learning algorithms, Robbins-Monro application, pattern recognition, classification algorithm, feature selection

1. Introduction

Most of the recent instances of pattern recognition problems (whether in computer vision, image understanding, biology, text interpretation, or spam detection) involve highly complex data sets with a huge number of possible explanatory variables. For many reasons, this abundance of variables significantly harms classification or recognition tasks. Weakly informative features act as artificial noise in data and limit the accuracy of classification algorithms. Also, the variance of a statistical model is typically an increasing function of the number of variables, whereas the bias is a decreasing function of this same quantity (Bias-Variance dilemma discussed by Geman et al., 1992); reducing the dimension of the feature space is necessary to infer reliable conclusions. There are efficiency issues, too, since the speed of many classification algorithms is largely improved when the complexity of the data is reduced. For instance, the complexity of the q-nearest neighbor algorithm varies proportionally with the number of variables. In some cases, the application of classification algorithms like Support Vector Machines (see Vapnik, 1998, 2000) or q-nearest neighbors on the full feature space is not possible or realistic due to the time needed to apply the decision rule. Also, there are many applications for which detecting the pertinent explanatory variables is critical, and as important as correctly performing classification tasks. This is the case, for example, in biology, where describing the source of a pathological state is equally important to just detecting it (Guyon et al., 2002; Golub et al., 1999).

Feature selection methods are classically separated into two classes. The first approach (*filter methods*) uses statistical properties of the variables to filter out poorly informative variables. This is done before applying any classification algorithm. For instance, singular value decomposition or independent component analysis of Jutten and Hérault (1991) remain popular methods to limit the dimension of signals, but these two methods do not always yield relevant selection of variables. In Bins and Draper (2001), superpositions of several efficient filters has been proposed to remove irrelevant and redundant features, and the use of a combinatorial feature selection algorithm has provided results achieving high reduction of dimensions (more than 80 % of features are removed) preserving good accuracy of classification algorithms on real life problems of image processing. Xing et al. (2001) have proposed a mixture model and afterwards an information gain criterion and a Markov Blanket Filtering method to reach very low dimensions. They next apply classification algorithms based on Gaussian classifier and Logistic Regression to get very accurate models with few variables on the standard database studied in Golub et al. (1999). The heuristic of Markov Blanket Filtering has been likewise competitive in feature selection for video application in Liu and Render (2003).

The second approach (*wrapper methods*) is computationally demanding, but often is more accurate. A wrapper algorithm explores the space of features subsets to optimize the induction algorithm that uses the subset for classification. These methods based on penalization face a combinatorial challenge when the set of variables has no specific order and when the search must be done over its subsets since many problems related to feature extraction have been shown to be NP-hard (Blum and Rivest, 1992). Therefore, automatic feature space construction and variable selection from a large set has become an active research area. For instance, in Fleuret and Geman (2001), Amit and Geman (1999), and Breiman (2001) the authors successively build tree-structured classifiers considering statistical properties like correlations or empirical probabilities in order to achieve good discriminant properties . In a recent work of Fleuret (2004), the author suggests to use mutual information to recursively select features and obtain performance as good as that obtained with a boosting algorithm (Friedman et al., 2000) with fewer variables. Weston et al. (2000) and Chapelle et al. (2002) construct another recursive selection method to optimize generalization ability with a gradient descent algorithm on the margin of Support Vector classifiers. Another effective approach is the Automatic Relevance Determination (ARD) used in MacKay (1992) which introduce a learning hierarchical prior over weights in a Bayesian Network, the weights connected to irrelevant features are automatically penalized which reduces their influence near zero. At last, an interesting work of Cohen et al. (2005) use an hybrid wrapper and filter approach to reach highly accurate and selective results. They consider an empirical loss function as a Shapley value and perform an iterative ranking method combined with backward elimination and forward selection. This last work is not so far from the ideas we develop in this paper.

In this paper, we provide an algorithm which attributes a weight to each feature, in relation with their importance for the classification task. The whole set of weights is optimized by a learning algorithm based on a training set. This weighting technique is not so new in the context of feature selection since it has been used in Sun and Li (2006). In our work, these weights result in an estimated probability distribution over features. This estimated probability will also be used to generate randomized classifiers, where the randomization is made on the variables rather than on the training set, an idea introduced by Amit and Geman (1997), and formalized by Breiman (2001). The selection algorithm and the randomized classifier will be tested on a series of examples.

The article is organized as follows. In Section 2, we describe our feature extraction model and the related optimization problem. Next, in Sections 3 and 4, we define stochastic algorithms which solve the optimization problem and discuss its convergence properties. In Section 5, we provide several applications of our method, first with synthetic data, then on image classification, spam detection and on microarray analysis. Section 6 is the conclusion and addresses future developments.

2. Feature Extraction Model

We start with some notations.

2.1 Primary Notation

We follow the general framework of supervised statistical pattern recognition: the input signal, belonging to some set $\mathcal{I}$, is modeled as a realization of a random variable, from which several computable quantities are accessible, forming a complete set of variables (or tests, or features) denoted $\mathcal{F} = \{\delta_1, \dots \delta_{|\mathcal{F}|}\}$. This set is assumed to be finite, although $|\mathcal{F}|$ can be a very large number, and our goal is to select the most useful variables. We denote $\delta(I)$ the complete set of features extracted from an input signal I.

A classification task is then specified by a finite partition $\mathcal{C} = \{C_1, \dots C_N\}$ of $\mathcal{I}$; the goal is to predict the class from the observation $\delta(I)$. For such a partition $\mathcal{C}$ and $I \in \mathcal{I}$, we write $\mathcal{C}(I)$ for the class C_i to which I belongs, which is assumed to be deterministic.

2.2 Classification Algorithm

We assume that a classification algorithm, denoted $\mathbb{A}$, is available, for both training and testing. We assume that $\mathbb{A}$ can be adapted to a subset $\omega \subset \mathcal{F}$ of *active variables* and to a specific classification task $\mathcal{C}$. (There may be some restriction on $\mathcal{C}$ associated to $\mathbb{A}$, for example, support vector machines are essentially designed for binary (two-class) problems.) In training mode, $\mathbb{A}$ uses a database to build an optimal classifier $\mathbb{A}_{\omega,\mathcal{C}} : \mathcal{I} \to \mathcal{C}$, such that $\mathcal{A}_{\omega,\mathcal{C}}(I)$ only depends on $\omega(I)$. We shall drop the subscript $\mathcal{C}$ when there is no ambiguity concerning the classification problem $\mathcal{C}$. The test mode simply consists in the application of $\mathcal{A}_\omega$ on a given signal.

We will work with a randomized version of $\mathbb{A}$, for which the randomization is with respect to the set of variables ω (see Amit and Geman, 1997; Breiman, 2001, 1998). In the training phase, this works as follows: first extract a collection $\{\omega^{(1)}, \dots, \omega^{(n)}\}$ of subsets of $\mathcal{F}$, and build the classifiers $\mathbb{A}_{\omega^{(1)}}, \dots, \mathbb{A}_{\omega^{(n)}}$. Then, perform classification in test phase using a majority rule for these n classifiers. This final algorithm will be denoted $\bar{\mathbb{A}}_{\mathcal{C}} = \bar{\mathbb{A}}(\omega^{(1)}, \dots, \omega^{(n)}, \mathcal{C})$. It is run with fixed $\omega^{(i)}$'s, which are obtained during learning.

Note that in this paper, we are not designing the classification algorithm, $\mathbb{A}$, for which we use existing procedures; rather, we focus on the extraction mechanism creating the random subsets of $\mathcal{F}$. This randomization process depends on the way variables are sampled from $\mathcal{F}$, and we will see that the design of a suitable probability on $\mathcal{F}$ for this purpose can significantly improve the classification rates. This probability therefore comes as a new parameter and will be learned from the training set.

Figure 1 summarizes the algorithm in test phase.

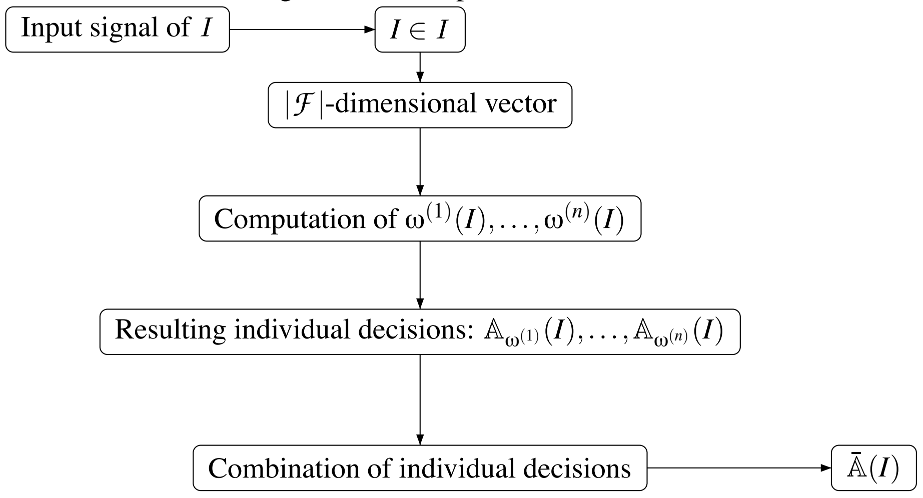

Input signal of I
$I \in \mathcal{I}$
$|\mathcal{F}|$-dimensional vector
Computation of $\omega^{(1)}(I), \dots, \omega^{(n)}(I)$
Resulting individual decisions: $\mathbb{A}_{\omega^{(1)}}(I), \dots, \mathbb{A}_{\omega^{(n)}}(I)$
Combination of individual decisions
$\bar{\mathbb{A}}(I)$

Figure 1: Global algorithm in test phase.

2.3 Measuring the Performance of the Algorithm $\mathbb{A}$

The algorithm $\mathbb{A}$ provides a different classifier $\mathbb{A}_\omega$ for each choice of a subset $\omega \subset \mathcal{F}$. We let q be the classification error: $q(\omega, \mathcal{C}) = \mathbf{P}(\mathbb{A}_\omega(I) \neq \mathcal{C}(I))$, which will be estimated by

$$q^\star(\omega, \mathcal{C}) = \hat{\mathbf{P}}(\mathbb{A}_\omega(I) \neq \mathcal{C}(I)) \tag{1}$$

where $\hat{\mathbf{P}}$ is the empirical probability on the training set.

We shall consider two particular cases for a given $\mathbb{A}$.

- Multi-class algorithm: assume that $\mathbb{A}$ is naturally adapted to multi-class problems (like a q-nearest neighbor, or random forest classifier). We then let $g(\omega) = q^\star(\omega, \mathcal{C})$ as defined above.
- Two-class algorithms: this applies to algorithms like support vector machines, which are designed for binary classification problems. We use the idea of the one-against-all method: denote by $\mathcal{C}_i$ the binary partition $\{C_i,\ \mathcal{I} \setminus C_i\}$. We then denote:

  $$g(\omega) = \frac{1}{N} \sum_{i=1}^{N} q^\star(\omega, \mathcal{C}_i)$$

  which is the average classification rate of the one vs. all classifiers. This "one against all strategy" can easily be replaced by others, like methods using error correcting output codes (see Dietterich and Bakiri, 1995).

2.4 A Computational Amendment

The computation $q^\star(\omega, \mathcal{C})$, as defined in Equation (1), requires training a new classification algorithm with variables restricted to ω, and estimating the empirical error; this can be rather costly with large data set (this has to be repeated at each of the steps of the learning algorithm).

Because of this, we use a slightly different evaluation of the error. In the algorithm, each time an evaluation of $q^{\star}(\omega, \mathcal{C})$ is needed, we use the following procedure (T being a fixed integer and $\mathcal{T}_{train}$ will be the training set):

1. Sample a subset $\mathcal{T}_1$ of size T (with replacements) from the training set.

2. Learn the classification algorithm on the basis of ω and $\mathcal{T}_1$.

3. Sample, with the same procedure, a subset $\mathcal{T}_2$ from the training set, and define $\hat{q}_{(\mathcal{T}_1,\mathcal{T}_2)}(\omega, \mathcal{C})$ to be the empirical error of the classifier learned *via* $\mathcal{T}_1$ on $\mathcal{T}_2$.

Since $\mathcal{T}_1$ and $\mathcal{T}_2$ are independent, we will use $\hat{q}(\omega, \mathcal{C})$ defined by

$$\boxed{\hat{q}(\omega, \mathcal{C}) = \mathbb{E}_{(\mathcal{T}_1,\mathcal{T}_2)}\left[\hat{q}_{(\mathcal{T}_1,\mathcal{T}_2)}(\omega, \mathcal{C})\right]}$$

to quantify the efficiency of the subset ω, where the expectation is computed over all the samples $\mathcal{T}_1, \mathcal{T}_2$ of signals taken from the training set of size T. It is also clear that defining such a cost function contributes in avoiding overfitting in the selection of variables. For the multiclass problem, we define

$$\hat{g}_{\mathcal{T}_1,\mathcal{T}_2}(\omega) = \frac{1}{N}\sum_{i=1}^{N} \hat{q}_{(\mathcal{T}_1,\mathcal{T}_2)}(\omega, \mathcal{C}_i)$$

and we replace the previous expression of g by the one below:

$$\boxed{g(\omega) = \mathbb{E}_{\mathcal{T}_1,\mathcal{T}_2}\left[\hat{g}_{\mathcal{T}_1,\mathcal{T}_2}(\omega)\right]}.$$

This modified function g will be used later in combination with a stochastic algorithm which will replace the expectation over the training and validation subsets by empirical averages. The selection of smaller training and validation sets for the evaluation of $\hat{g}_{\mathcal{T}_1,\mathcal{T}_2}$ then represents a huge reduction of computer time. The selection of the size of $\mathcal{T}_1$ and $\mathcal{T}_2$ depends on the size of the original training set and of the chosen learning machine. It has so far been selected by hand.

In the rest of our paper, the notation $\mathbb{E}_{\xi}[.]$ will refer to the expectation using ξ as the integration variable.

2.5 Weighting the Feature Space

To select a group a variables which are most relevant for the classification task one can think first of a *hard selection method*, that is, search $\underline{\omega}$ such that

$$\hat{q}(\underline{\omega}, \mathcal{C}) = \underset{\omega \in \mathcal{F}^{|\omega|}}{\arg\min}\, \hat{q}(\omega, \mathcal{C}).$$

But sampling all possible subsets (ω covers $\mathcal{F}^{|\omega|}$) may be untractable since $|\mathcal{F}|$ can be thousands and $|\omega|$ ten or hundreds.

We address this with a soft selection procedure that attributes weights to the features $\mathcal{F}$.

2.6 Feature Extraction Procedure

Consider a probability distribution $\mathbb{P}$ on the set of features $\mathcal{F}$. For an integer k, the distribution $\mathbb{P}^{\otimes k}$ corresponds to k independent trials with distribution $\mathbb{P}$. We define the cost function $\mathcal{E}$ by

$$\mathcal{E}(\mathbb{P}) = \mathbb{E}_{\mathbb{P}^{\otimes k}} g(\omega) = \sum_{\omega \in \mathcal{F}^k} g(\omega)\mathbb{P}^{\otimes k}(\omega). \tag{2}$$

Our goal is to minimize this averaged error rate with respect to the selection parameter, which is the probability distribution $\mathbb{P}$. The relevant features will then be the set of variables $\delta \in \mathcal{F}$ for which $\mathbb{P}(\delta)$ is large. The global (iterative) procedure that we propose for estimating $\mathbb{P}$ is summarized in Figure 2.

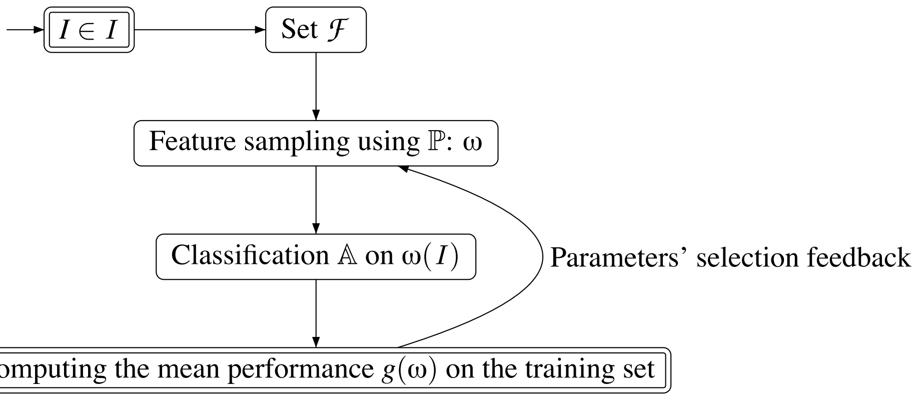

$I \in I$
Set $\mathcal{F}$
Feature sampling using $\mathbb{P}$: ω
Classification $\mathbb{A}$ on $\omega(I)$
Parameters' selection feedback
Computing the mean performance $g(\omega)$ on the training set

Figure 2: Scheme of the procedure for learning the probability $\mathbb{P}$.

Remark: We use $\mathbb{P}$ here as a control parameter: we first make a random selection of features before setting the learning algorithm. It is thus natural to optimize our way to select features from $\mathcal{F}$ and formalize it as a probability distribution on $\mathcal{F}$. The number of sampled features (k) is a hyper-parameter of the method. Although we have set it by hand in our experiments, it can be estimated by cross-validation during learning.

3. Search Algorithms

We describe in this section three algorithmic options to minimize the energy $\mathcal{E}$ with respect to the probability $\mathbb{P}$. This requires computing the gradient of $\mathcal{E}$, and dealing with the constraints implied by the fact that $\mathbb{P}$ must be a probability distribution. These constraints are summarized in the following notation.

We denote by $\mathcal{S}_{\mathcal{F}}$ the set of all probability measures on $\mathcal{F}$: a vector $\mathbb{P}$ of $\mathbb{R}^{|\mathcal{F}|}$ belongs to $\mathcal{S}_{\mathcal{F}}$ if

$$\sum_{\delta \in \mathcal{F}} \mathbb{P}(\delta) = 1 \tag{3}$$

and

$$\forall \delta \in \mathcal{F} \quad \mathbb{P}(\delta) \geq 0. \tag{4}$$

We also denote $\mathcal{H}_{\mathcal{F}}$ the hyperplane in $\mathbb{R}^{|\mathcal{F}|}$ which contains $\mathcal{S}_{\mathcal{F}}$, defined by (3).

We define the projections of an element of $\mathbb{R}^{|\mathcal{F}|}$ onto the closed convex sets $\mathcal{S}_{\mathcal{F}}$ and $\mathcal{H}_{\mathcal{F}}$. Let $\pi_{\mathcal{S}_{\mathcal{F}}}(x)$ be the closest point of $x \in \mathbb{R}^{|\mathcal{F}|}$ in $\mathcal{S}_{\mathcal{F}}$

$$\pi_{\mathcal{S}_{\mathcal{F}}}(x) = \arg\min_{y \in \mathcal{S}_{\mathcal{F}}} \left\{ \|y - x\|_2{}^2 \right\}$$

and similarly

$$\pi_{\mathcal{H}_{\mathcal{F}}}(x) = \arg\min_{y \in \mathcal{H}_{\mathcal{F}}} \left\{ \|y - x\|_2{}^2 \right\} = x - \frac{1}{|\mathcal{F}|} \sum_{\delta \in \mathcal{F}} x(\delta).$$

The latter expression comes from the fact that the orthogonal projection of a vector x onto a hyperplane is $x - \langle x, N \rangle N$ where N is the unit normal to the hyperplane. For $\mathcal{H}_{\mathcal{F}}$, N is the vector with all coordinates equal to $1/\sqrt{|\mathcal{F}|}$.

Our first option will be to use projected gradient descent to minimize $\mathcal{E}$, taking only constraint (3) into account. This implies solving the gradient descent equation

$$\frac{d\mathbb{P}_t}{dt} = -\pi_{\mathcal{H}_{\mathcal{F}}}(\nabla \mathcal{E}(\mathbb{P}_t)) \tag{5}$$

which is well-defined as long as $\mathbb{P}_t \in \mathcal{S}_{\mathcal{F}}$. We will also refer to the discretized form of (5),

$$\mathbb{P}_{n+1} = \mathbb{P}_n - \varepsilon_n \pi_{\mathcal{H}_{\mathcal{F}}}(\nabla \mathcal{E}(\mathbb{P}_n)) \tag{6}$$

with positive $(\varepsilon_n)_{n \in \mathbb{N}}$. Again, this equation can be implemented as long as $\mathbb{P}_n \in \mathcal{S}_{\mathcal{F}}$. We will later propose two new strategies to deal with the positivity constraint (4), the first one using the change of variables $\mathbb{P} \mapsto \log \mathbb{P}$, and the second being a constrained optimization algorithm, that we will implement as a constrained stochastic diffusion on $\mathcal{S}_{\mathcal{F}}$.

3.1 Computation of the Gradient

However, returning to (5), our first task is to compute the gradient of the energy. We first do it in the standard case of the Euclidean metric on $\mathcal{S}_{\mathcal{F}}$, that is we compute $\nabla_{\mathbb{P}} \mathcal{E}(\delta) = \partial \mathcal{E} / \partial \mathbb{P}(\delta)$. For $\omega \in \mathcal{F}^k$ and $\delta \in \mathcal{F}$, denote by $C(\omega, \delta)$ the number of occurrences of δ in ω:

$$\boxed{C(\omega, \delta) = |\{i \in \{1, \ldots k\} \quad | \quad \omega_i = \delta\}|}.$$

$C(\omega, .)$ is then the $|\mathcal{F}|$-dimensional vector composed by the set of values $C(\omega, \delta), \delta \in \mathcal{F}$. Then, a straightforward computation gives:

Proposition 1 *If $\mathbb{P}$ is any point of $\mathcal{S}_{\mathcal{F}}$, then*

$$\forall \delta \in \mathcal{F} \qquad \nabla_{\mathbb{P}} \mathcal{E}(\delta) = \sum_{\omega \in \mathcal{F}^k} \frac{C(\omega, \delta) \mathbb{P}^{\otimes k}(\omega)}{\mathbb{P}(\delta)} g(\omega). \tag{7}$$

Consequently, the expanded version of (6) is

$$\mathbb{P}_{n+1}(\delta) = \mathbb{P}_n(\delta) - \varepsilon_n \sum_{\omega \in \mathcal{F}^k} \mathbb{P}^{\otimes k}(\omega) g(\omega) \left(\frac{C(\omega, \delta)}{\mathbb{P}(\delta)} - \frac{1}{|\mathcal{F}|} \sum_{\mu \in \omega} \frac{C(\omega, \mu)}{\mathbb{P}(\mu)} \right). \tag{8}$$

In the case when $\mathbb{P}(\delta) = 0$, then, necessarily $C(\omega, \delta) = 0$ and the term $C(\omega, \delta)/\mathbb{P}(\delta)$ is by convention equal to 0.

The positivity constraint is not taken in account here, but this can be dealt with, as described in the next section, by switching to an exponential parameterization. It is also be possible to design a constrained optimization algorithm, exploring the faces of the simplex when needed, but this is a rather complex procedure, which is harder to conciliate with the stochastic approximations we will describe in Section 4. This approach will in fact be computationally easier to handle with a constrained stochastic diffusion algorithm, as described in Section 3.3.

3.2 Exponential Parameterization and Riemannian Gradient

Define $y(\delta) = \log \mathbb{P}(\delta)$ and

$$\mathcal{Y} = \left\{ y = (y(\delta), \delta \in \mathcal{F}) \mid \sum_{\delta \in \mathcal{F}} e^{y(\delta)} = 1 \right\}$$

which is in one-to-one correspondence with $\mathcal{S}_{\mathcal{F}}$ (allowing for the choice $y(\delta) = -\infty$). Define

$$\tilde{\mathcal{E}}(y) = \mathcal{E}(\mathbb{P}) = \sum_{\omega \in \mathcal{F}^k} e^{y(\omega_1) + \cdots + y(\omega_k)} g(\omega).$$

Then, we have:

Proposition 2 *The gradient of $\mathcal{E}$ with respect to these new variables is given by:*

$$\nabla_y \tilde{\mathcal{E}}(\delta) = \sum_{\omega \in \mathcal{F}^k} e^{y(\omega_1) + \cdots + y(\omega_k)} C(\omega, \delta) g(\omega). \tag{9}$$

We can interpret this gradient on the variables y as a gradient on the variables $\mathbb{P}$ with a Riemannian metric

$$\langle u, v \rangle_{\mathbb{P}} = \sum_{\delta \in \mathcal{F}} u(\delta) v(\delta) / \mathbb{P}(\delta).$$

The geometry of the space $\mathcal{S}_{\mathcal{F}}$ with this metric D has the property that the boundary points $\partial \mathcal{S}_{\mathcal{F}}$ are at infinite distance from any point into the interior of $\mathcal{S}_{\mathcal{F}}$.

Denoting $\tilde{\nabla}$ for the gradient with respect to this metric, we have in fact, with $y = \log \mathbb{P}$:

$$\tilde{\nabla}_{\mathbb{P}} \mathcal{E}(\delta) = \nabla_y \tilde{E} = \sum_{\omega \in \mathcal{F}^k} \mathbb{P}^{\otimes k}(\omega) C(\omega, \delta) g(\omega).$$

To handle the unit sum constraint, we need to project this gradient on the tangent space to $\mathcal{Y}$ at point y. Denoting this projection by π_y, we have

$$\pi_y(w) = w - \langle w | e^y \rangle / \| e^y \|^2$$

where e^y is the vector with coordinates $e^{y(\delta)}$. This yields the evolution equation in the y variables

$$\frac{dy_t(\delta)}{\mathrm{d}t} = -\nabla_{y_t} \tilde{\mathcal{E}}(\delta) + \kappa_t e^{y_t(\delta)}, \tag{10}$$

where

$$\kappa_t = \left(\sum_{\delta' \in \mathcal{F}} \nabla_{y_t} \tilde{\mathcal{E}}(\delta') e^{y_t(\delta')}\right) / \left(\sum_{\delta' \in \mathcal{F}} e^{2y_t(\delta')}\right)$$

does not depend on δ.

The associated evolution equation for $\mathbb{P}$ becomes

$$\frac{d\mathbb{P}_t(\delta)}{\mathrm{d}t} = -\mathbb{P}_t(\delta)\left(\tilde{\nabla}_{\mathbb{P}_t}\mathcal{E}(\delta) - \kappa_t \mathbb{P}_t(\delta)\right). \tag{11}$$

Consider now a discrete time approximation of (11), under the form

$$\mathbb{P}_{n+1}(\delta) = \frac{\mathbb{P}_n(\delta)}{K_n} e^{-\varepsilon_n\left(\tilde{\nabla}_{\mathbb{P}_n}\mathcal{E}(\delta) - \kappa_n \mathbb{P}_n(\delta)\right)} \tag{12}$$

where the newly introduced constant K_n ensures that the probabilities sum to 1. This provides an alternative scheme of gradient descent on $\mathcal{E}$ which has the advantage of satisfying the positivity constraints (4) by construction.

- Start with $\mathbb{P}_0 = \mathcal{U}_{\mathcal{F}} \longmapsto y_0 = \log \mathbb{P}_0$,
- Until convergence: Compute $\mathbb{P}_{n+1}$ from Equation (12).

Remark: In terms of y_n, (12) yields

$$y_{n+1}(\delta) = y_n(\delta) - \varepsilon_n\left(\tilde{\nabla}_{\mathbb{P}_n}\mathcal{E}(\delta) - \kappa_n \mathbb{P}_n(\delta)\right) - \log K_n .$$

The definition of the constant K_n implies that

$$K_n = \sum_{\delta \in \mathcal{F}} \mathbb{P}_n e^{-\varepsilon_n\left(\tilde{\nabla}_{\mathbb{P}_n}\mathcal{E}(\delta) - \kappa_n \mathbb{P}_n(\delta)\right)}.$$

We can write a second order expansion of the above expression to deduce that

$$K_n = \sum_{\delta \in \mathcal{F}} \mathbb{P}_n(\delta) - \varepsilon_n \mathbb{P}_n(\delta)\left(\tilde{\nabla}_{\mathbb{P}_n}\mathcal{E}(\delta) - \kappa_n \mathbb{P}_n(\delta)\right) + A_n \varepsilon_n^2 = 1 + A_n \varepsilon_n^2$$

since, by definition of κ_n:

$$\sum_{\delta \in \mathcal{F}} \mathbb{P}_n(\delta)(\tilde{\nabla}_{\mathbb{P}_n} E(\delta) - \kappa_n \mathbb{P}_n(\delta)) = 0.$$

Consequently, there exists a constant B which depends on k and $\max(\varepsilon_n)$ such that, for all n, $|\log K_n| \leq B\varepsilon_n^2$.

3.3 Constrained Diffusion

The algorithm (8) can be combined with reprojection steps to provide a consistent procedure. We implement this using a stochastic diffusion process constrained to the simplex $\mathcal{S}_{\mathcal{F}}$. The associated stochastic differential equation is

$$\mathrm{d}\mathbb{P}_t = -\nabla_{\mathbb{P}_t}\mathcal{E}\mathrm{d}t + \sqrt{\sigma}\mathrm{d}W_t + \mathrm{d}Z_t$$

where $\mathcal{E}$ is the cost function introduced in (2), σ is a positive non-degenerate matrix on $\mathcal{H}_{\mathcal{F}}$ and dZ_t is a stochastic process which accounts for the jumps which appear when a reprojection is needed. In other words, $d|Z_t|$ is positive if and only if $\mathbb{P}_t$ hits the boundary $\partial\mathcal{S}_{\mathcal{F}}$ of our simplex.

The rigorous construction of such a process is linked to the theory of Skorokhod maps, and can be found in works of Dupuis and Ishii (1991) and Dupuis and Ramanan (1999). Existence and uniqueness are true under general geometric conditions which are satisfied here.

4. Stochastic Approximations

The evaluation of $\nabla\mathcal{E}$ in the previous algorithms requires summing the efficiency measures $g(\omega)$ over all ω in $\mathcal{F}^k$. This is, as already discussed, an untractable sum. This however can be handled using a stochastic approximation, as described in the next section.

4.1 Stochastic Gradient Descent

We first recall general facts on stochastic approximation algorithms.

4.1.1 Applying the ODE Method

Stochastic approximations can be seen as noisy discretizations of deterministic ODE's (see Benveniste et al., 1990; Benaïm, 2000; Duflo, 1996). They are generally expressed under the form

$$X_{n+1} = X_n + \varepsilon_n F(X_n, \xi_{n+1}) + \varepsilon_n^2 \eta_n \tag{13}$$

where X_n is the current state of the process, ξ_{n+1} a random perturbation, and η_n a secondary error term. If the distribution of ξ_{n+1} only depends on the current value of X_n (Robbins-Monro case), then one defines an average drift $X \mapsto G(X)$ by

$$G(X) = \mathbb{E}_{\xi}[F(X, \xi)]$$

and the Equation (13) can be shown to evolve similarly to the ODE $\dot{X} = G(X)$, in the sense that the trajectories coincide when $(\varepsilon_n)_{n\in\mathbb{N}}$ goes to 0 (a more precise statement is given in Section 4.1.4).

4.1.2 Approximation Terms

To implement our gradient descent equations in this framework, we therefore need to identify two random variables d_n or $\tilde{d}_n$ such that

$$\mathbb{E}[d_n] = \pi_{\mathcal{H}_{\mathcal{F}}}[\nabla_{\mathbb{P}_n}\mathcal{E}] \qquad \text{and} \qquad \mathbb{E}\left[\tilde{d}_n\right] = \pi_{y_n}\left[\nabla_{y_n}\tilde{\mathcal{E}}\right]. \tag{14}$$

This would yield the stochastic algorithm:

$$\mathbb{P}_{n+1} = \mathbb{P}_n - \varepsilon_n d_n \text{ or } \mathbb{P}_{n+1} = \mathbb{P}_n \frac{e^{-\varepsilon_n \tilde{d}_n}}{K_n}.$$

¿From (7), we have:

$$\nabla_{\mathbb{P}}\mathcal{E}(\delta) = \mathbb{E}_{\omega}\left[\frac{C(\omega,\delta)g(\omega)}{\mathbb{P}(\delta)}\right].$$

Using the linearity of the projection $\pi_{\mathcal{H}_\mathcal{F}}$, we get

$$\pi_{\mathcal{H}_\mathcal{F}}\left(\nabla\mathcal{E}(\mathbb{P})\right)(\delta)=\mathbb{E}_\omega\left[\pi_{\mathcal{H}_\mathcal{F}}\left(\frac{C(\omega,.)g(\omega)}{\mathbb{P}(.)}\right)(\delta)\right].$$

Consequently, following (14), it is natural to define the approximation term of the gradient descent (5) by:

$$\boxed{d_n=\pi_{\mathcal{H}_\mathcal{F}}\left(\frac{C(\omega_n,.)\hat{q}_{\mathcal{T}_1^n,\mathcal{T}_2^n}(\omega_n,\mathcal{C})}{\mathbb{P}_n(.)}\right)} \tag{15}$$

where the set of k features ω_n is a random variable extracted from $\mathcal{F}$ with law $\mathbb{P}_n^{\otimes k}$ and $\mathcal{T}_1^n,\mathcal{T}_2^n$ are independently sampled into the training set $\mathcal{T}$.

In a similar way, we can compute the approximation term of the gradient descent based on (9) since

$$\nabla_y\tilde{\mathcal{E}}(\delta)=\mathbb{E}_\omega\left[g(\omega)C(\omega,\delta)\right]$$

yielding

$$\tilde{d}_n=\pi_{y_n}\left(C(\omega_n,.)\hat{q}_{\mathcal{T}_1^n,\mathcal{T}_2^n}(\omega_n,\mathcal{C}))\right)$$

where π_y is the projection on the tangent space $\mathcal{T}\mathcal{Y}$ to the sub-manifold $\mathcal{Y}$ at point y, and ω_n is a random variable extracted from $\mathcal{F}$ with the law $\mathbb{P}_n^{\otimes k}$.

By construction, we therefore have the proposition

Proposition 3 *The mean effect of random variables d_n and $\tilde{d}_n$ is the global gradient descent, in other words:*

$$\mathbb{E}\left[d_n\right]=\pi_{\mathcal{H}_\mathcal{F}}\left(\nabla\mathcal{E}(\mathbb{P}_n)\right)$$

and

$$\mathbb{E}\left[\tilde{d}_n\right]=\pi_{y_n}\left(\nabla\tilde{\mathcal{E}}(y_n)\right).$$

4.1.3 LEARNING THE PROBABILITY MAP $(\mathbb{P}_n)_{n\in\mathbb{N}}$

We now make explicit the learning algorithms for Equations (5) and (10). We recall the definition of

$$C(\omega,\delta)=\left|\{i\in\{1,\dots k\}\quad|\quad\omega_i=\delta\}\right|$$

where δ is a given feature and ω a given feature subset of length k which is an hyperparameter (see bottom of page 6). $\hat{q}_{\mathcal{T}_1,\mathcal{T}_2}(\omega,\mathcal{C})$, which is the empirical classification error on $\mathcal{T}_2$ for a classifier trained on $\mathcal{T}_1$ using features in ω.

• Euclidean gradient (Figure 3):

• Riemannian Gradient: For the Riemannian case, we have to give few modifications for the update step (Figure 4).

The mechanism of the two former algorithms summarized by Figures 3,4 can be intuitively explained looking carefully at the update step. For instance, in the first case, at step n, one can see that for all features of $\delta\in\omega_n$, we substract from $\mathbb{P}_n(\delta)$ amount proportional to the error performed with ω and inversely proportional to $\mathbb{P}_n(\delta)$ although for other features out of ω_n, weights are a little bit increased. Consequently, worst features with poor error of classification will be severely decreased, particularly when they are suspected to be bad (small weight $\mathbb{P}_n$).

```
Let $\mathcal{F} = (\delta_1, \dots \delta_{|\mathcal{F}|})$, integers $\mu, T$ and a real number $\alpha$ (stoping criterion)
$n = 0$: define $\mathbb{P}_0$ to be the uniform distribution $\mathcal{U}_\mathcal{F}$ on $\mathcal{F}$.
While $\left|\mathbb{P}_{n-\lfloor n/\mu \rfloor} - \mathbb{P}_n\right|_\infty > \alpha$ and $\mathbb{P}_n \geq 0$:
        Extract $\omega_n$ with replacement from $\mathcal{F}^k$ with respect to $\mathbb{P}_n^{\otimes k}$.
        Extract $\mathcal{T}_1^n$ and $\mathcal{T}_2^n$ of size $T$ with uniform independent samples over $\mathcal{T}_{train}$.
        Compute $\hat{q}_{\mathcal{T}_1^n, \mathcal{T}_2^n}(\omega_n, \mathcal{C})$and the drift vector $d_n$ where
            $d_n(\delta) = \hat{q}_{\mathcal{T}_1^n, \mathcal{T}_2^n}(\omega_n, \mathcal{C}) \left( \frac{C(\omega_n, \delta)}{\mathbb{P}_n(\delta)} - \sum_{\mu \in \omega_n} \frac{C(\omega_n, \mu)}{|\mathcal{F}| \mathbb{P}_n(\mu)} \right).$
        Update $\mathbb{P}_{n+1}$ with $\mathbb{P}_{n+1} = \mathbb{P}_n - \varepsilon_n . d_n$.
        $n \mapsto n+1$.
```

Figure 3: Euclidean gradient Algorithm.

Remark We provide the Euclidean gradient algorithm, which is subject to failure (one weight might become nonpositive) because it may converge for some applications, and in these cases, is much faster than the exponential formulation.

4.1.4 Convergence of the Approximation Scheme

This more technical section can be skipped without harming the understanding of the rest of the paper. We here rapidly describe in which sense the stochastic approximations we have designed converge to their homologous ODE's. This is a well-known fact, especially in the Robbins-Monro case that we consider here, and the reader may refer to works of Benveniste et al. (1990); Duflo (1996); Kushner and Yin (2003), for more details. We follow the terminology employed in the approach of Benaim (1996).

Fix a finite dimensional open set E. A differential flow $(t, x) \mapsto \phi_t(x)$ is a time-indexed family of diffeomorphisms satisfying the semi-group condition $\phi_{t+h} = \phi_h \circ \phi_t$ and $\phi_0 = Id$; $\phi_t(x)$ is typically given as the solution of a differential equation $\frac{dy}{dt} = G(y)$, at time t, with initial condition $y(0) = x$. Asymptotic pseudotrajectories of a differential flow are defined as follows:

```
Let $\mathcal{F} = (\delta_1, \dots \delta_{|\mathcal{F}|})$, integers $\mu, T$ and a real number $\alpha$ (stoping criterion)
$n = 0$: define $\mathbb{P}_0$ to be the uniform distribution $\mathcal{U}_\mathcal{F}$ on $\mathcal{F}$.
While $\left|\mathbb{P}_{n-\lfloor n/\mu \rfloor} - \mathbb{P}_n\right|_\infty > \alpha$:
        Extract $\omega_n$ from $\mathcal{F}^k$ with respect to $\mathbb{P}_n^{\otimes k}$.
        Extract $\mathcal{T}_1^n$ and $\mathcal{T}_2^n$ of size $T$ with uniform independent samples over $\mathcal{T}_{train}$.
        Compute $\hat{q}_{\mathcal{T}_1^n, \mathcal{T}_2^n}(\omega_n, \mathcal{C})$.
        Update $\mathbb{P}_{n+1}$ with:
            $\mathbb{P}_{n+1}(\delta) = \frac{\mathbb{P}_n(\delta) e^{-\varepsilon_n (C(\omega_n, \delta) \hat{q}_{\mathcal{T}_1^n, \mathcal{T}_2^n}(\omega_n, \mathcal{C}) + \kappa_n \mathbb{P}_n(\delta))}}{K_n}$ with
            $\kappa_n = \left(\sum_{\delta' \in \omega_n} C(\omega_n, \delta') \mathbb{P}_n(\delta')\right) / \left(\left(\sum_{\delta' \in \mathcal{F}} \mathbb{P}_n(\delta')^2\right)\right.$
        and $K_n$ is a normalization constant.
        $n \mapsto n+1$
```

Figure 4: Riemannian gradient Algorithm.

Definition 4 *A map $X : \mathbb{R}^+ \longmapsto E$ is an asymptotic pseudotrajectory of the flow ϕ if, for all positive numbers T*

$$\lim_{t\mapsto\infty} \sup_{0\leq h\leq T} \|X(t+h) - \phi_h(X(t))\| = 0.$$

In other words, the tails of the trajectory X asymptotically coincides, within any finite horizon T, with the flow trajectories.

Consider algorithms of the form

$$X_{n+1} = X_n + \varepsilon_n F(X_n, \xi_{n+1}) + \varepsilon_n^2 \eta_{n+1}$$

with $X_n \in E$, ξ_{n+1} a first order perturbation (such that the conditional distribution knowing all present and past variables only depends on X_n), and η_n a secondary noise process. The variable X_n can be linearly interpolated into a time-continuous process as follows: define $\tau_n = \sum_{k=1}^{n} \varepsilon_k$ and $X_{\tau_n} = X_n$; then let X_t be linear and continuous between τ_n and τ_{n+1}, for $n \geq 0$.

Consider the mean ODE

$$\frac{dx}{dt} = G(x) = \mathbb{E}_{\xi}[F(X,\xi)|X = x]$$

and its associated flow ϕ. Then, under mild conditions on F and η_n, and under the assumption that $\sum_{n>0} \varepsilon_n^{1+\alpha} < \infty$ for some $\alpha > 0$, the linearly interpolated process X_t is an asymptotic pseudotrajectory of ϕ. We will consequently choose $\varepsilon_n = \varepsilon/(n+C)$ where ε and C are positive constants fixed at start of our algorithms. We can here apply this result with $X_n = y_n$, $\xi_{n+1} = (\omega_n, \mathcal{T}_1^n, \mathcal{T}_2^n)$ and $\eta_{n+1} = \log K_n/\varepsilon_n^2$ for which all the required conditions are satisfied since for the Euclidean case, when $\omega_n \sim \mathbb{P}_n^{\otimes k}$ and $(\mathcal{T}_1, \mathcal{T}_2) \sim \mathcal{U}_T^{\otimes 2T}$:

$$\begin{aligned}
\mathbb{E}_{\omega_n}[F(\mathbb{P}_n, \omega_n)] &= \mathbb{E}_{\omega_n, \mathcal{T}_1, \mathcal{T}_2}[d_n(\omega_n, \mathcal{T}_1, \mathcal{T}_2)] \\
&= \mathbb{E}_{\omega_n, \mathcal{T}_1, \mathcal{T}_2}\left[\pi_{\mathcal{H}_\mathcal{F}}\left(\frac{C(\omega_n, .)\hat{q}_{\mathcal{T}_1, \mathcal{T}_2}(\omega_n, \mathcal{C})}{\mathbb{P}_n(.)}\right)\right] \\
&= \mathbb{E}_{\omega_n}\left[\pi_{\mathcal{H}_\mathcal{F}}\left(\frac{C(\omega_n, .)\mathbb{E}_{\mathcal{T}_1, \mathcal{T}_2}\left[\hat{q}_{\mathcal{T}_1, \mathcal{T}_2}(\omega_n, \mathcal{C})\right]}{\mathbb{P}_n(.)}\right)\right] \\
&= \mathbb{E}_{\omega_n}\left[\pi_{\mathcal{H}_\mathcal{F}}\left(\frac{C(\omega_n, .)\hat{q}(\omega_n, \mathcal{C})}{\mathbb{P}_n(.)}\right)\right] \\
\mathbb{E}_{\omega_n}[F(\mathbb{P}_n, \omega_n)] &= \pi_{\mathcal{H}_\mathcal{F}}\left(\nabla\mathcal{E}(\mathbb{P}_n)\right).
\end{aligned}$$

4.2 Numerical Simulation of the Diffusion Model

We use again (15) for the approximation of the gradient of $\mathcal{E}$. The theoretical basis for the convergence of this type of approximation can be found in Buche and Kushner (2001) and Kushner and Yin (2003), for example. A detailed convergence proof is provided in Gadat (2004).

This results in the following numerical scheme. We recall the definition of

$$C(\omega, \delta) = |\{i \in \{1, \dots k\} \quad | \quad \omega_i = \delta\}|$$

and of $\hat{q}_{\mathcal{T}_1,\mathcal{T}_2}(\omega,\mathcal{C})$,which is the empirical classification error on $\mathcal{T}_2$ for a classifier trained on $\mathcal{T}_1$ using features in ω.

```
Let $\mathcal{F} = (\delta_1, \dots \delta_{|\mathcal{F}|})$, an integer
$n = 0$: define $\mathbb{P}_0$ to be the uniform distribution $\mathcal{U}_{\mathcal{F}}$ on $\mathcal{F}$.
Iterate the loop:
        Extract $\omega_n$ from $\mathcal{F}^k$ with respect to $\mathbb{P}_n^{\otimes k}$.
        Extract $\mathcal{T}_1^n$ and $\mathcal{T}_2^n$ of size $T$ with uniform independent samples over $\mathcal{T}_{train}$.
        Compute $\hat{q}_{\mathcal{T}_1^n,\mathcal{T}_2^n}(\omega_n,\mathcal{C})$.
        Compute the intermediate state $\mathbb{Q}_n$ (may be out of $\mathcal{S}_{\mathcal{F}}$):
                $\mathbb{Q}_n = \mathbb{P}_n - \varepsilon_n \frac{C(\omega_n,.)\hat{q}_{\mathcal{T}_1^n,\mathcal{T}_2^n}(\omega_n)}{\mathbb{P}_n} + \sqrt{\varepsilon_n}\sqrt{\sigma} d\xi_n$
                where $d\xi_n$ is a centered normal $|\mathcal{F}|$ dimensional vector.
        Project $\mathbb{Q}_n$ on $\mathcal{S}_{\mathcal{F}}$ to obtain $\mathbb{P}_{n+1}$:
                $\mathbb{P}_{n+1} = \pi_{\mathcal{S}_{\mathcal{F}}}(\mathbb{Q}_n) = \mathbb{Q}_n + dz_n$.
        $n \mapsto n+1$.
```

Figure 5: Constrained diffusion.

4.3 Projection on $\mathcal{S}_{\mathcal{F}}$

The natural projection on $\mathcal{S}_{\mathcal{F}}$ can be computed in a finite number of steps as follows.

1. Define $X^0 = X$, if X^0 does not belong to the hyperplane $\mathcal{H}_{\mathcal{F}}$, project first X^0 to $\mathcal{H}_{\mathcal{F}}$:

$$X^1 = \pi_{\mathcal{H}_{\mathcal{F}}}(X^0).$$

2. – If X^k belongs to $\mathcal{S}_{\mathcal{F}}$, stop the recursion.
   – Else, call J^k the set of integers i such that $X_i^k \leq 0$ and define X^{k+1} by

$$\forall i \in J^k \qquad X_i^{k+1} = 0.$$

$$\forall i \notin J^k \qquad X_i^{k+1} = X_i^k + \frac{1}{|\mathcal{F}| - |J^k|}\left(1 - \sum_{j \notin J^k} X_j^k\right).$$

One can show that the former recursion stops in at most $|\mathcal{F}|$ steps (see Gadat, 2004, chap. 4).

5. Experiments

This section provides a series of experimental results using the previous algorithms. Table 1 briefly summarizes the parameters of the several experiments performed.

5.1 Simple Examples

We start with a simple, but illustrative, small dimensional example.

Data Set	Dim.	$\mathbb{A}$	Classes	Training Set	Test Set
Synthetic	100	N.N.	3	500	100
IRIS	4	CART	3	100	50
Faces	1926	SVM	2	7000	23000
SPAM	54	N.N.	2	3450	1151
USPS	2418	SVM	10	7291	2007
Leukemia	3859	SVM	2	72	$\emptyset$
ARCENE	10000	SVM	2	100	100
GISETTE	5000	SVM	2	6000	1000
DEXTER	20000	SVM	2	300	300
DOROTHEA	100000	SVM	2	800	350
MADELON	500	N.N.	2	2000	600

Table 1: Characteristics of the data sets used in experiments.

5.1.1 SYNTHETIC EXAMPLE

Data We consider $|\mathcal{F}| = 100$ ternary variables and 3 classes (similar results can be obtained with more classes and variables). We let $\mathcal{I} = \{-1;0;1\}^f$ and the feature $\delta_i(I)$ simply be the ith coordinate of $I \in \mathcal{I}$. Let $\mathcal{G}$ be a subset of $\mathcal{F}$. We define the probability distribution $\mu(\,;\mathcal{G})$ in $\mathcal{I}$ to be the one for which all δ in $\mathcal{F}$ are independent, $\delta(I)$ follows a uniform distribution on $\{-1;0;1\}$ if $\delta \notin \mathcal{G}$ and $\delta(I) = 1$ if $\delta \in \mathcal{G}$. We model each class by a mixture of such distributions, including a small proportion of noise. More precisely, for a class C_i, $i = 1,2,3$, we define

$$\mu_i(I) = \frac{q}{3}\left(\mu(I;\mathcal{F}_i^1) + \mu(I;\mathcal{F}_i^2) + \mu(I;\mathcal{F}_i^3)\right) + (1-q)\mu(I;\emptyset)$$

with $q = 0.9$ and

$$\mathcal{F}_1^1 = \{\delta_1;\delta_3;\delta_5;\delta_7\}, \qquad \mathcal{F}_1^2 = \{\delta_1;\delta_5\}, \qquad \mathcal{F}_1^3 = \{\delta_3;\delta_7\},$$

$$\mathcal{F}_2^1 = \{\delta_2;\delta_4;\delta_6;\delta_8\}, \qquad \mathcal{F}_2^2 = \{\delta_2;\delta_4\}, \qquad \mathcal{F}_2^3 = \{\delta_6;\delta_8\},$$

$$\mathcal{F}_3^1 = \{\delta_1;\delta_4;\delta_8;\delta_9\}, \qquad \mathcal{F}_3^2 = \{\delta_1;\delta_8\}, \qquad \mathcal{F}_3^3 = \{\delta_4;\delta_9\}.$$

In other words, these synthetic data are generated with almost deterministic values on some variables (which depends on the class the sample belongs to) and with a uniform noise elsewhere. We expect our learning algorithm to put large weights on features in $\mathcal{F}_i^j$ and ignore the other ones. The algorithm $\mathbb{A}$ we use in this case is an M nearest neighbour classification algorithm, with distance given by

$$d(I_1,I_2) = \sum_{\delta\in\omega} \chi_{\delta(I_1)\neq\delta(I_2)}.$$

This toy example is interesting because it is possible to compute the exact gradient of $\mathcal{E}$ for small values of M and $k = |\omega|$. Thus, we can compare the stochastic gradient algorithms with the exact gradient algorithm and evaluate the speed of decay of $\mathcal{E}$. Moreover, one can see in the construction of our signals that some features are relevant with several classes (reusables features), some features are important only for one class and others are simply noise on the input. This will permit to evaluate the model of "frequency of goodness" used by OFW.

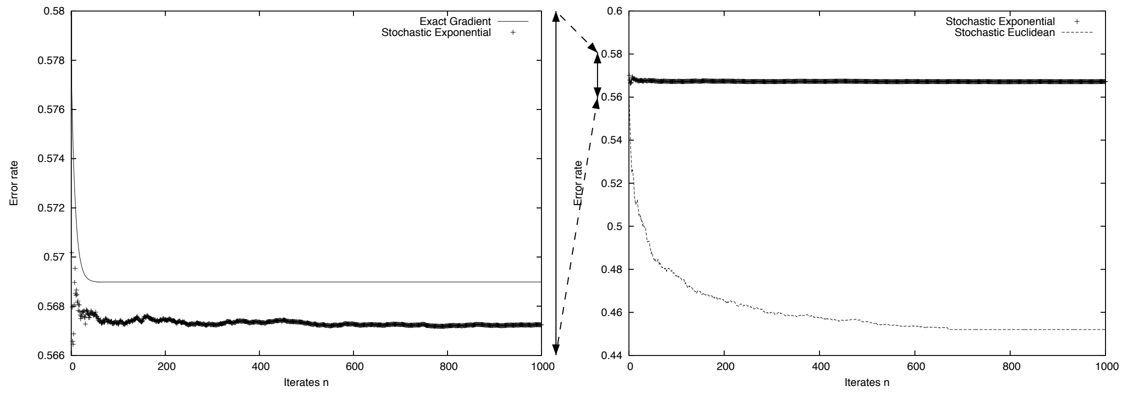


Figure 6: Note that the left and right figures are drawn on different scales. Left: Exact gradient descent (full line) vs. stochastic exponential gradient descent (dashed line) classification rates on the training set. Right: Stochastic Euclidean gradient descent (full line) vs. stochastic exponential gradient descent (dashed line) classification rates on the training set.

Results We provide in Figure 6 the evolution of the mean error $\mathcal{E}$ on our training set set against the computation time for exact and stochastic exponential gradient descent algorithms. The exact algorithm is faster but is quickly captured in a local minimum although exponential stochastic descent avoids more traps. Also shown in Figure 6, is the fact that the stochastic Euclidean method achieved better results faster than the exponential stochastic approach and avoided more traps than the exponential algorithm to reach lower error rates.

Note that Figure 6 (and similar plots in subsequent experiments) is drawn for the comparison of the numerical procedures that have been designed to minimize the training set errors. This does not relate to the generalization error of the final classifier, which is evaluated on test sets.

Finally, Figure 7 shows that the efficiency of the stochastic gradient descent and of the reflected diffusion are almost similar in our synthetic example. This has in fact always been so in our experiments: the diffusion is slightly better than the gradient when the latter converges. For this reason, we will only compare the exponential gradient and the diffusion in the experiments which follow. Finally, we summarize this instructive synthetic experiments in Figure 8. Remark that in this toy example; the exact gradient descent and the Euclidean stochastic gradient (first algorithm of Section 4.1.3) are almost equivalent.

In Figure 9, we provide the probabilities of the first 15 features in function of $k = |\omega|$. (The graylevel is proportional to the probability obtained at the limit).

Interpretation We observe that the features which are preferably selected are those which lie in several subspaces $\mathcal{F}_i^j$, and which bring information for at least two classes. These are *reusable features*, the knowledge of which being very precious information for the understanding of pattern recognition problems. This result can be compared to selection methods based on information theory. One simple method is to select the variables which provide the most information to the class, and therefore minimize the conditional entropy (see Cover and Thomas, 1991) of the class given each variable. In this example, this conditional entropy is 1.009 for features contained in none of the sets $\mathcal{F}_i^j$, 0.852 for those contained in only one set and approximatively 0.856 for those

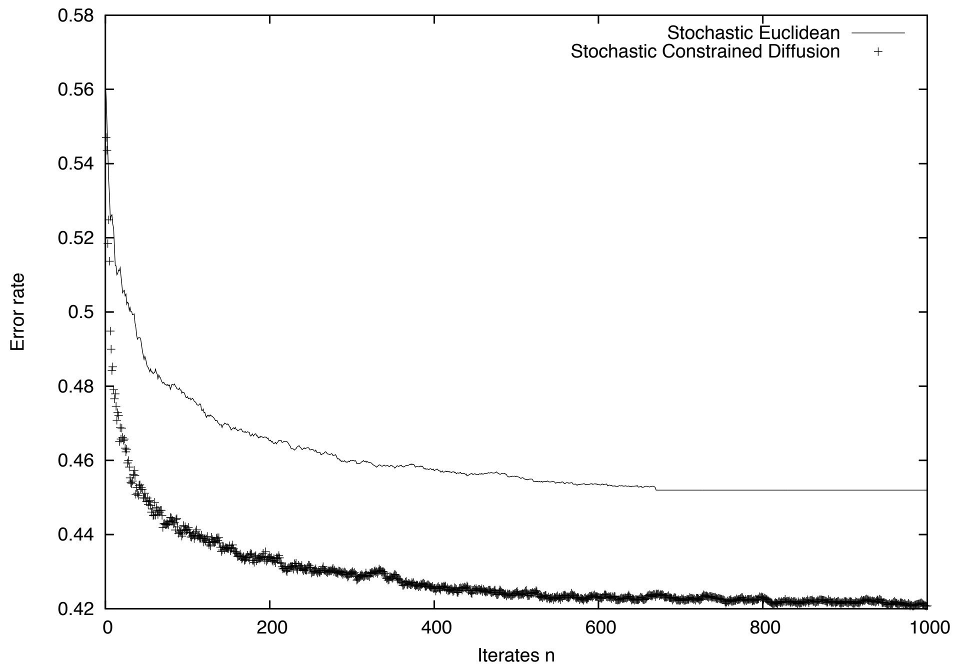


Figure 7: Stochastic Euclidean gradient descent (dashed line) vs. reflected diffusion (full line) classification rate on the training set.

contained in two of these sets. This implies that this information-based criterion would correctly discard the non-informative variables, but fail to discriminate between the last two groups.

Remark finally that the features selected by OFW after the reusables ones are still relevant for the classification task.

5.1.2 IRIS DATABASE

We use in this section the famous Fisher's IRIS database where data are described by the 4 variables: "Sepal Length", "Sepal Width", "Petal Length" and "Petal Width". Even though our framework is to select features in a large dictionary of variables, it will be interesting to look at the behavior of our algorithm on IRIS since results about feature selection are already known on this classical example. We use here a Classification and Regression Tree (CART) using the Gini index. We extract 2 variables at each step of the algorithm, 100 samples out of 150 are used to train our feature weighting procedure. The Figures 10 and 11 describe the behavior of our algorithms (with and without the noise term).

We remark here that for each one of our two approaches, we approximately get the entire weight on the last two variables " Petal Length" (70%) and "Petal Width" (30%). This result is consistent with the selection performed by CART on this database since we obtain similar results as seen in Figure 12. Moreover, a selection based on the Fisher score reaches the same results for this very simple and low dimensional example.

The classification on Test Set is improved selecting two features (with OFW as Fisher Scoring) since we obtain an error rate of 2.6% although without any selection, CART provides an error rate

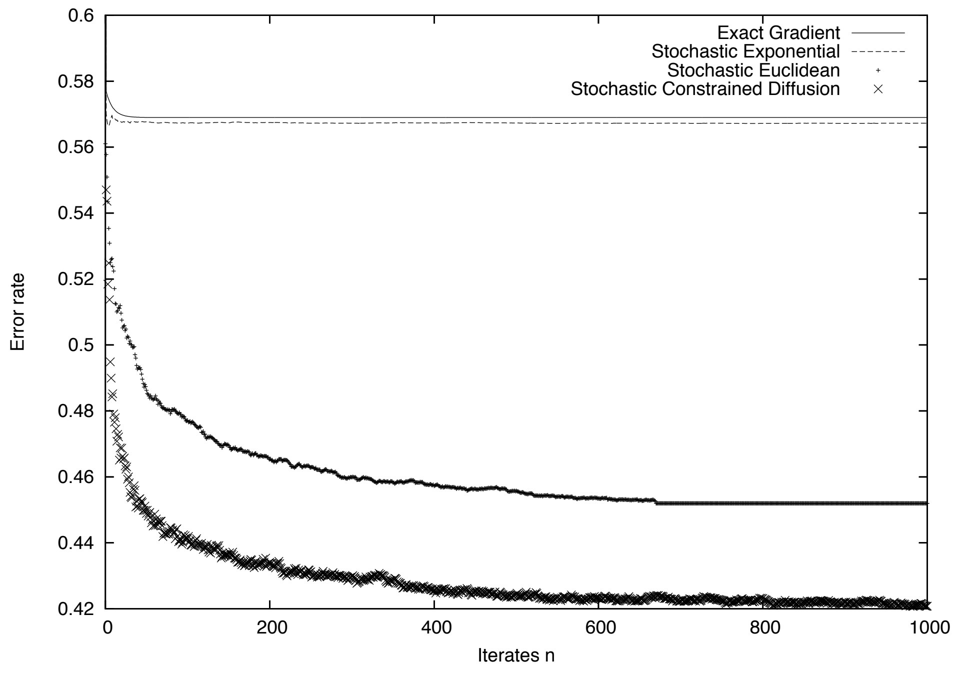


Figure 8: Comparison of the mean error rate computed on the test set with the 4 exact or stochastics gradient descents.

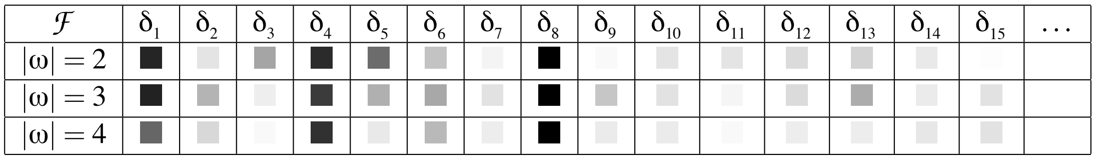

$\mathcal{F}$
δ_1
δ_2
δ_3
δ_4
δ_5
δ_6
δ_7
δ_8
δ_9
δ_{10}
δ_{11}
δ_{12}
δ_{13}
δ_{14}
δ_{15}
...
$|\omega| = 2$
$|\omega| = 3$
$|\omega| = 4$

Figure 9: Probability histogram for several values of $|\omega|$.

of 4%. In this small low dimensional example, OFW quickly converges to the optimal weight and we obtain a ranking coherent with the selection performed by Fisher Score or CART.

5.2 Real Classification Problems

We now address real pattern recognition problems. We also compare our results with other algorithms: no selection method, Fisher scoring method, Recursive Feature Elimination method (RFE) of Guyon et al. (2002), L0-Norm for linear support vector machines of Weston et al. (2003) and Random Forests (RF) of Breiman (2001). We used for these algorithms Matlab implementations provided by the Spider package for RFE and L0-SVM;[1] and the random forest package of Leo Breiman.[2] In our experiments, we arbitrarily fixed the number of features per classifier (to 100 for the Faces, Handwritten Digits and Leukemia data and to 15 for the email database). It would be possible to also optimize it, through cross-validation, for example, once the optimal $\mathbb{P}$ has been

1. This package is available at http://www.kyb.tuebingen.mpg.de/bs/people/spider/main.html.
2. Codes are available on http://www.stat.berkeley.edu/users/breiman/RandomForests.

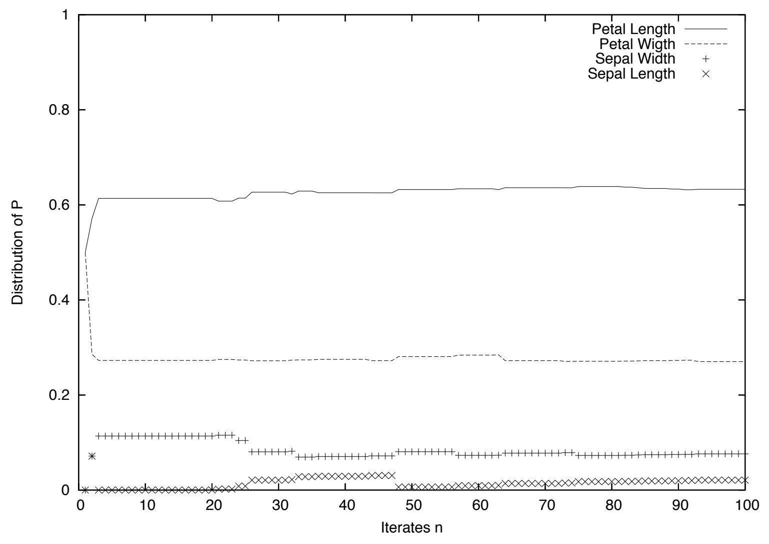


Figure 10: Evolution with *n* of the distribution on the 4 variables using a stochastic Euclidean algorithm.

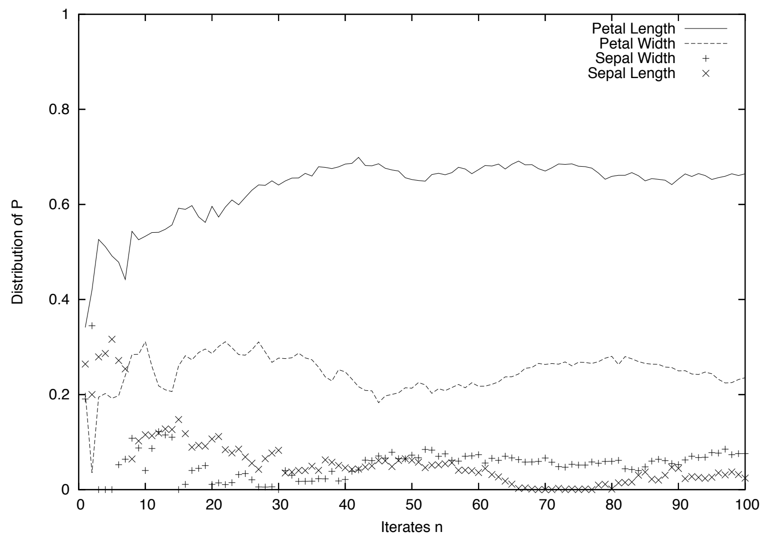


Figure 11: Evolution with *n* of the distribution on the 4 variables using a stochastic Euclidean diffusion algorithm.

computed (running this optimization online, while also estimating $\mathbb{P}$ would be too computationally intensive). We have remarked in our experiments that the estimation of $\mathbb{P}$ was fairly robust to to variations of the number of features extracted at each step (*k* in our notation). In particular, taking *k* too large does not help much.

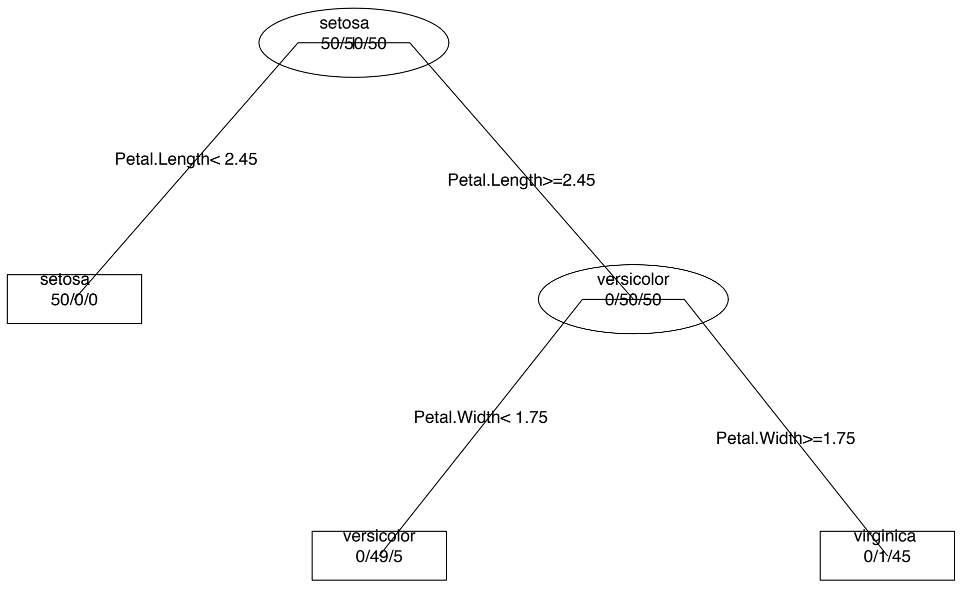


Figure 12: Complete classification tree of IRIS generated from recursive partitioning (CART implementation is using the rpart library of R).

5.2.1 FACE DETECTION

Experimental framework We use in this section the face database from MIT, which contains 19×19 gray level images; samples extracted from the database are represented in Figure 13. The database contains almost 7000 images to train and more than 23000 images to test.

The features in $\mathcal{F}$ are binary edge detectors, as developed in works of Amit and Geman (1999); Fleuret and Geman (2001). This feature space has been shown to be efficient for classification in visual processing. We therefore have as many variables and dimensions as we have possible edge detectors on images. We perform among the whole set of these edge detectors a preprocessing step described in Fleuret and Geman (2001). We then obtain 1926 binary features, each one defined by its orientation, vagueness and localisation.

The classification algorithm $\mathbb{A}$ which is used here is an optimized version of Linear Support Vector Machines developed by Joachims and Klinkenberg (2000); Joachims (2002) (with linear kernel).

Results We first show the improvement of the mean performance of our extraction method, learned on the training set, and computed on the test set, from a random uniform sampling of features (Figure 14).

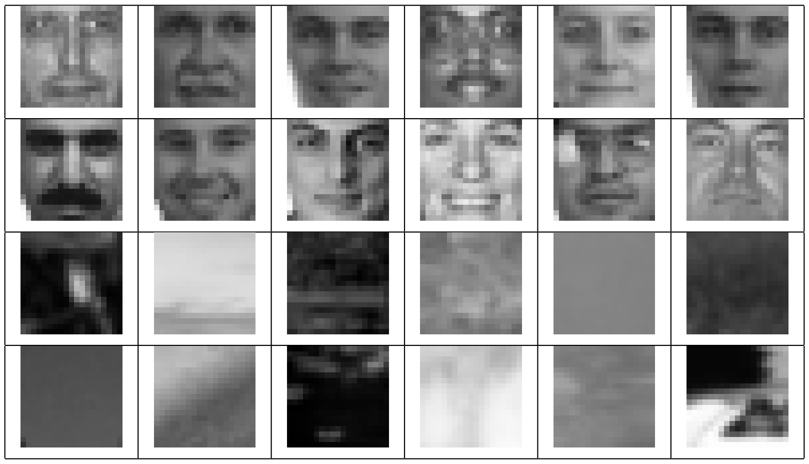
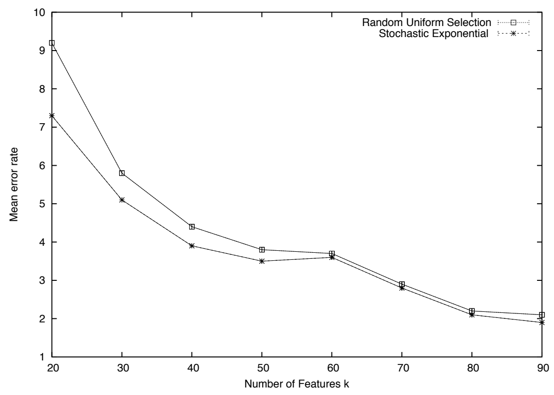


Figure 13: Sample of images taken from MIT database.



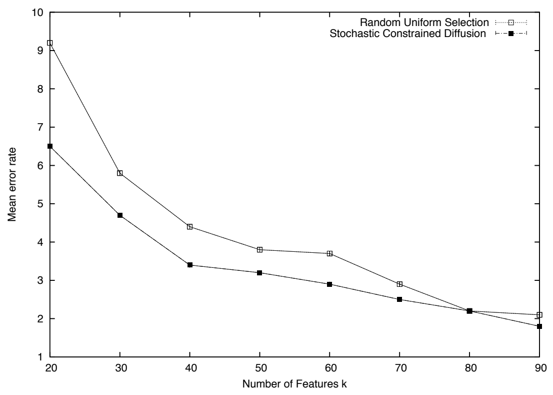


Figure 14: Left: Evolution with k of the average classification error of faces recognition on the test set using a uniform law (dashed line) and $\mathbb{P}_\infty$ (full line), learned with a stochastic gradient method with exponential parameterization. Right: same comparison, for the constrained diffusion algorithm.

Our feature extraction method based on learning the distribution $\mathbb{P}$ improves significantly the classification rate, particularly for weak classifiers ($k = 20$ or 30 for example) as shown in Figure 14. We remark again that the constrained diffusion performs better than the stochastic exponential gradient. We achieve a 1.6% error rate after learning with a reflected diffusion, or 1.7% with a stochastic exponential gradient (2% before learning). The analysis of the most likely features (which are the most weighted variables) is also interesting, and occurs in meaningful positions, as shown in Figure 15.

Figure 16 shows a comparison of the efficiency (computed on the test set) of Fisher, RFE, L0-SVM and our weighting procedure to select features; besides we have shown the performance of $\mathbb{A}$ without any selection and the best performance of Random Forests (as an asymptote).

We observe that our method is here less efficient with a small number of features (for 20 features selected, we obtain 7.3% while RFE and L0 selections get 4% and 3.2% of misclassification rate).

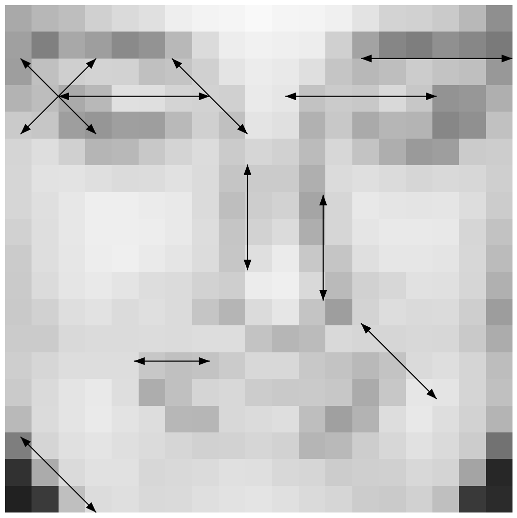


Figure 15: Representation of the main edge detectors after learning.

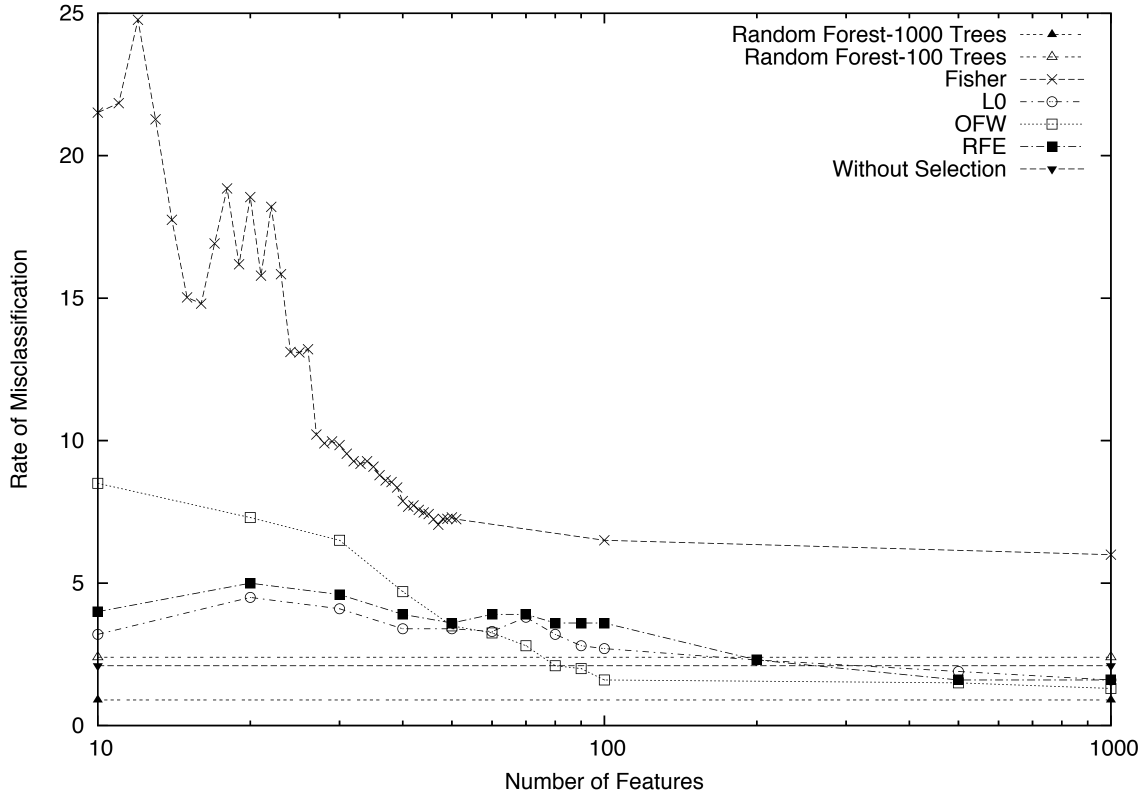


Figure 16: Efficiency of several feature extractions methods for the faces database test set.

However, for a larger set of features, our weighting method is more effective than other methods since we obtained 1.6% of misclassification for 100 features selected (2.7% for L0 selection and 3.6% for RFE).

The comparison with the Random Forest algorithm is more difficult to estimate: one tree achieves 2.4% error but the length of this tree is more than 1000 and this error rate is obtained by the 3 former algorithms using only 200 features. The final best performance on this database is obtained using Random Forests with 1000 trees. We obtain then a misclassification rate of 0.9%.

5.2.2 SPAM CLASSIFICATION

Experimental framework This example uses a text database available at D.J. Newman and Merz (1998), which contains emails received by a research engineer from the HP labs, and divided into SPAM and non SPAM categories. The features here are the rates of appearance of some keywords (from a list of 57) in each text. As the problem is quite simple using the last 3 features of the previous list, we choose to remove these 3 variables (which depends on the number of capital letters in an email), we start consequently with a list of 54 features. We use here a 4-nearest neighbor algorithm and we extract 15 features at each step. The database is composed by 4601 messages and we use 75% of the email database to learn our probability $\mathbb{P}_\infty$, representing our extraction method while the 25% samples of data is left to form the test set.

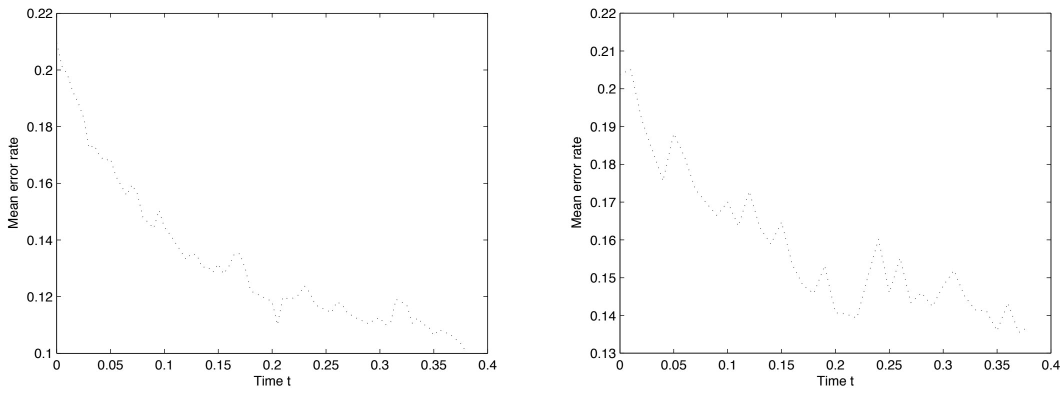


Figure 17: Time evolution of the energy $\mathcal{E}_1$ for the spam/email classification D.J. Newman and Merz (1998) computed on the test set, using a stochastic gradient descent with an exponential parameterization (left) and with a constrained diffusion (right).

Results We plot the average error on the test set in Figure 17. On our test set, the method based on the exponential parameterization achieves better results than those obtained by reflected diffusion which is slower because of our Brownian noise. The weighting method is here again efficient in improving the performances of the Nearest Neighbor algorithm.

Moreover, we can analyze the words selected by our probability $\mathbb{P}_\infty$. In the next table, two columns provide the features that are mainly selected. We achieve in a different way similar results to those noticed in Hastie et al. (2001) regarding the ranking importance of the words used for spam detection.

The words which are useful for spam recognition (left columns) are not surprising ("business", "remove", "receive" or "free" are highly weighted). More interesting are the words selected in the right column; these words are here useful to enable a personal email detection. Personal informa-

Words favored for SPAM	Frequency	Words favored for NON SPAM	Frequency
remove	8.8%	cs	5.4%
business	8.7%	857	4.6%
[	6%	415	4.4%
report	5.9%	project	4.3%
receive	5.6%	table	4.2%
internet	4.4%	conference	4.2%
free	4.1%	lab	3.9%
people	3.7%	labs	3.2%
000	3.6%	edu	2.8%
direct	2.3%	650	2.7%
!	1.2%	85	2.5%
$	1%	george	1.6%

Figure 18: Words mostly selected by $\mathbb{P}_\infty$ (exponential gradient learning procedure) for the spam/email classification.

tions like phone numbers ("650", "857") or first name ("george") are here favored to detect real email messages. The database did not provide access to the original messages, but the importance of the phone numbers or first name is certainly due to the fact that many non-spam messages are replies to previous messages outgoing from the mailbox, and would generally repeat the original sender's signature, including its first name, address and phone number.

We compare next the performances obtained by our method with RFE, RF and L0-SVM. Figure 19 show relative efficiency of these algorithms on the spam database.

Without any selection, the linear SVM has more than 15% error rate while each one of the former feature selection algorithms achieve better results using barely 5 words. The best algorithm is here the L0-SVM method, while the performance of our weighting method (7.47% with 20 words) is located between RFE (11.1% with 20 words) and L0-SVM (4.47% with 20 words). In addition, RF high performance is obtained using a small forest of 5 trees (not as deep as in the example of faces recognition) and we obtain with this algorithm 7.24% of misclassification rate using trees of size varying from 50 to 60 binary tests.

5.2.3 HANDWRITTEN NUMBER RECOGNITION

Experimental framework A classical benchmark for pattern recognition algorithms is the classification of handwritten numbers. We have tested our algorithm on the USPS database (Hastie et al., 2001; Schölkopf and Smola, 2002): each image is a segment from a ZIP code isolating a single digit. The 7291 images of the training set and 2007 of the test set are 16×16 eight-bit grayscale maps, with intensity between 0 and 255. We use the same feature set, $\mathcal{F}$, as in the faces example. We obtain a feature space $\mathcal{F}$ of 2418 edge detectors with one precise orientation, location and blurring parameter. The classification algorithm $\mathbb{A}$ we used is here again a linear support vector machine.

Results Since our reference wrapper algorithms (RFE and L0-SVM) are restricted to 2 class problems, we present only results obtained on this database with the algorithm $\mathbb{A}$ which is a SVM based on the "one versus all" idea.

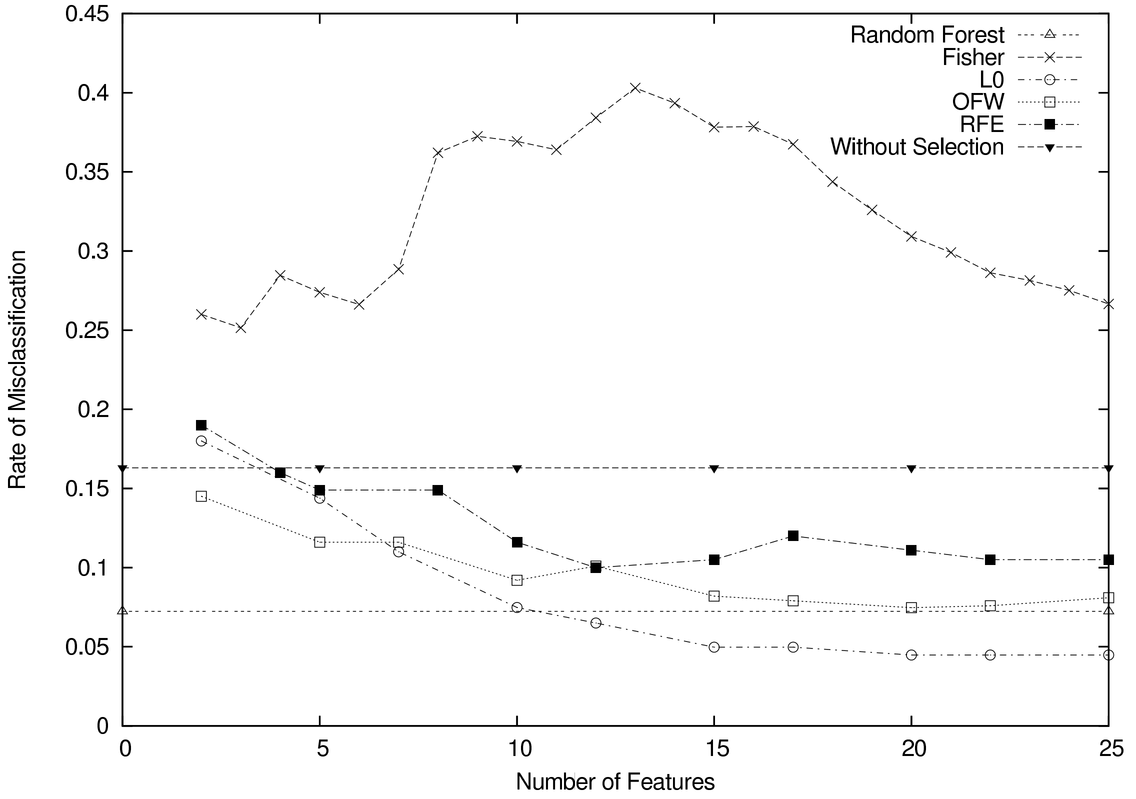


Figure 19: Efficiency of several feature extractions methods on the test set for the SPAM database.

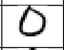
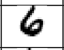
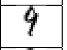
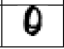
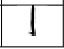
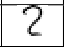
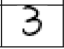
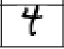
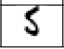
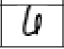
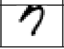
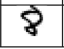
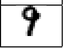
Class	$\mathcal{C}_0$	$\mathcal{C}_1$	$\mathcal{C}_2$	$\mathcal{C}_3$	$\mathcal{C}_4$	$\mathcal{C}_5$	$\mathcal{C}_6$	$\mathcal{C}_7$	$\mathcal{C}_8$	$\mathcal{C}_9$
Image I										
										

Figure 20: Sample of images taken from the USPS database.

The improvement of the detection rate is also similar to the previous example, as shown in Figure 21. We first plot the mean classification error rate before and after learning the probability map $\mathbb{P}$. These rates are obtained by averaging $g(\omega)$ over samples of features uniformly distributed on $\mathcal{F}$ in the first case, and distributed according to $\mathbb{P}$ in the second case. These numbers are computed on *training data* and therefore serve for evaluation of the efficiency of the algorithm in improving the energy function from $\mathcal{E}_1(\mathcal{U}_{\mathcal{F}})$ to $\mathcal{E}_1(\mathbb{P}_\infty)$. Figure 21 provides the variation of the mean error rate in function of the number of features k used in each ω. The ratio between the two errors (before and after learning) rates, is around 90% independently on the value of k.

Figure 22 provides the result of the classification algorithm (using the voting procedure) on the test set. The majority vote is based on 10 binary SVM-classifiers on each binary classification problem C_i vs. $I \setminus C_i$. The features are extracted first with the uniform distribution $\mathcal{U}_{\mathcal{F}}$ on $\mathcal{F}$, then using the final $\mathbb{P}_\infty$.

The learning procedure significantly improves the performance of the classification algorithms. The final average error rate on the USPS database is about 3.2% for 10 elementary classifiers per

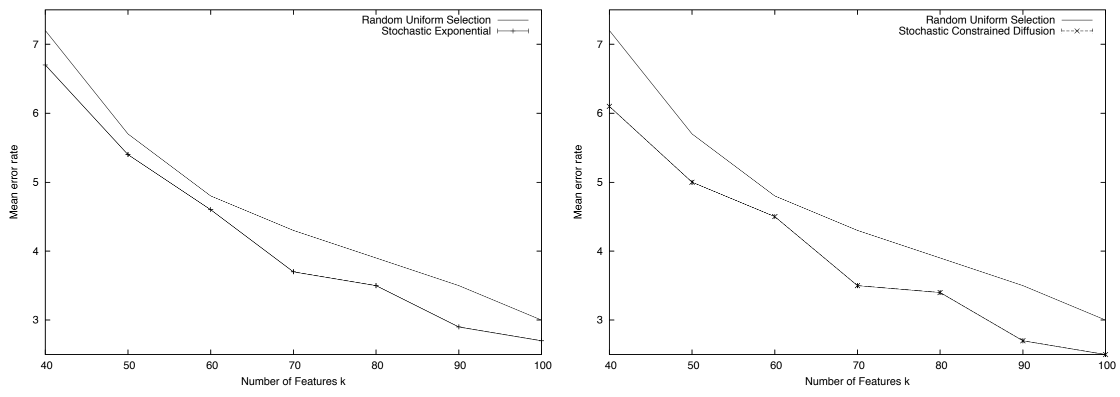


Figure 21: Mean error rate over the training set USPS for k varying from 40 to 100, before (dashed line) and after (full line) a stochastic gradient learning based on exponential parameterization (left) and constrained diffusion (right).

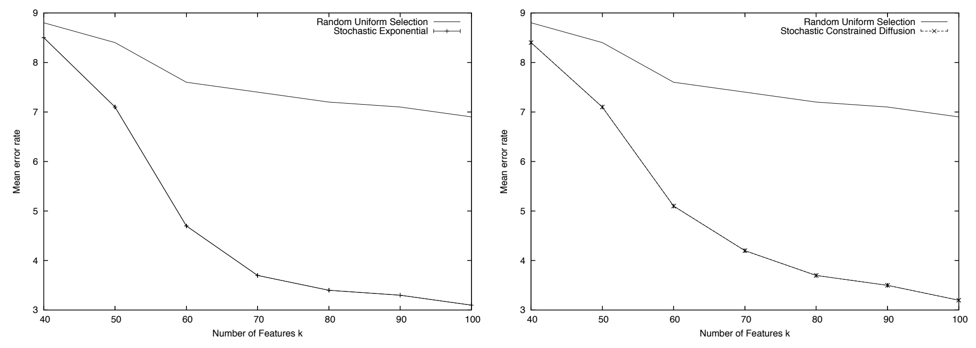


Figure 22: Evolution with k of the mean error of classification on the test set, extraction based on random uniform selection (dashed line) and $\mathbb{P}_\infty$ selection (full line) for USPS data, learning computed with stochastic gradient using exponential parameterization (left) and constrained diffusion (right).

class $\mathcal{C}_i$, with 100 binary features per elementary classifier. The performance is not as good as the one obtained by the tangent distance method of Simard and LeCun (1998) (2.7% error rate of classification), but we here use very simple (edge) features. And the result is better, for example, than linear or polynomial Support Vector Machines (8.9% and 4% error rate) computed without any selection and than sigmoid kernels (4.1%) (see Schölkopf et al., 1995) with a reduced complexity (measured, for example by the needed amount of memory).

Since the features we consider can be computed at every location in the image, it is interesting to visualize where the selection has occurred. This is plotted in Figure 23, for the four types of edges we consider (horizontal, vertical and two diagonal), with grey levels proportional to the value of $\mathbb{P}_\infty$ for each feature.

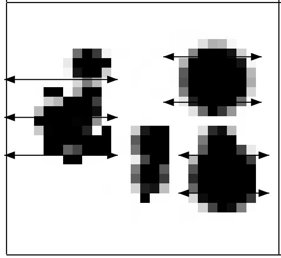
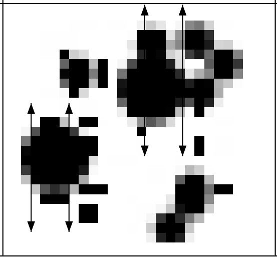
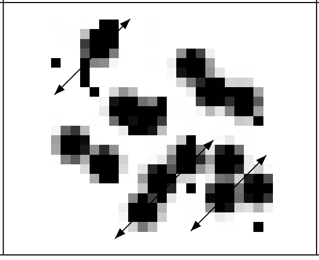
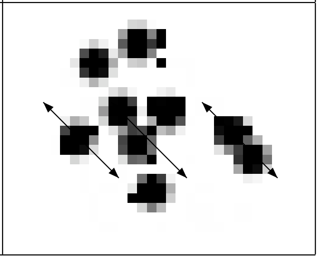
Horizontal edges	Vertical edges	Diagonal edges “$+\pi/4$”	Diagonal edges “$-\pi/4$”
			

Figure 23: Representation of the selected features after a stochastic exponential gradient learning for USPS digits. Greyscales are proportional to weights of features

5.2.4 GENE SELECTION FOR LEUKEMIA *AML-ALL* RECOGNITION

Experimental framework We carry on our experiments with feature selection and classification for microarray data. We have used the Leukemia Database AML-ALL of Golub et al. (1999). We have a very small number of samples (72 signals) described by a very large number of genes. We run a preselection method to obtain the database used by Deb and Reddy (2003) that contains 3859 genes.[3] Our algorithm $\mathbb{A}$ is here a linear support vector machines. As we face a numerical problem with few samples on each class (AML and ALL), we decide to benchmark each of the algorithms we have tested using a 10-fold cross validation method.

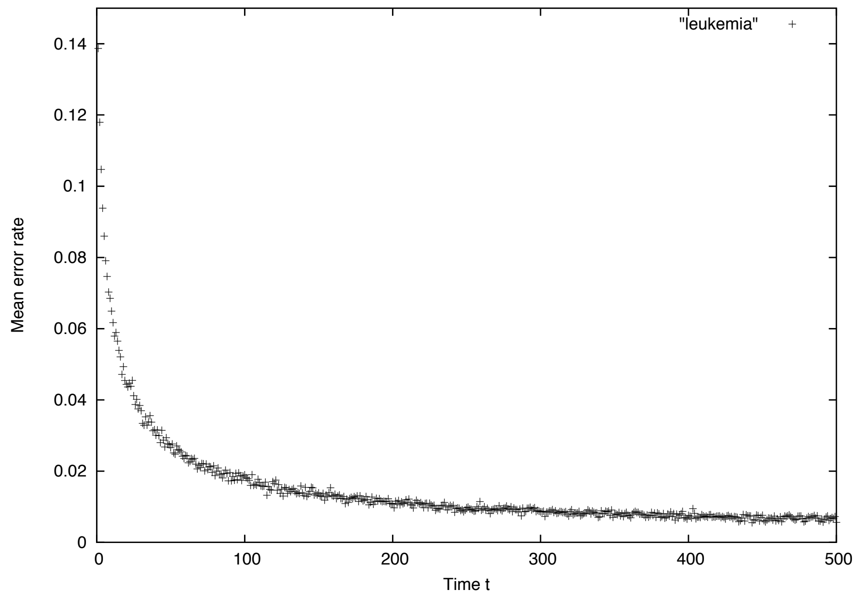


Figure 24: Evolution of the mean energy $\mathcal{E}$ computed by the constrained diffusion method on the training set with time for $k = 100$.

3. Data Set is available on http://www.lsp.ups-tlse.fr/Fp/Gadat/leukemia.txt.

Results Figure 24 shows the efficiency of our method of weighting features in reducing the mean error $\mathcal{E}$ on the training set. We remark that with random uniform selection of 100 features, linear support vector machines obtain a poor rate larger than 15% while learning $\mathbb{P}_\infty$, we achieve a mean error rate less than 1%.

We now compare our result to RFE, RF and L0-SVM using the 10-fold cross validation method. Figure 25 illustrates this comparison between these former algorithms. In this example, we obtain

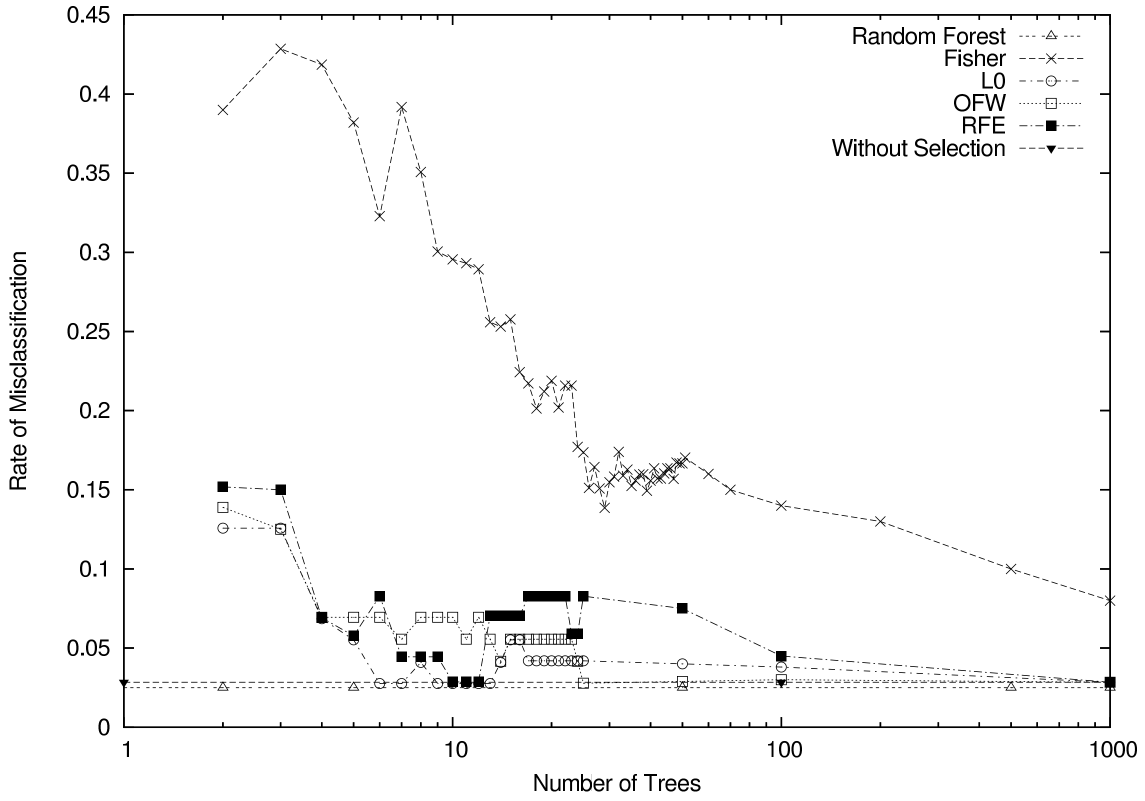


Figure 25: Efficiency of several feature extractions methods for the Leukemia database. Performances are computed using 10 CV.

better results without any selection, but in fact the classification of one linear SVM does not permit to rank features by importance effect on the classification task. We note here again that our weighting method is less effective for short size subsets (5 genes) while our method is competitive with larger subsets (20-25 genes). Here again, we note that L0-SVM outperforms RFE (like in the SPAM study Section 5.2.2). Finally, the Random Forest algorithm obtains results which are very irregular in connection with the number of trees as one can see in Figure 26.

5.2.5 FEATURE SELECTION CHALLENGE

We conclude our experiments with results on the Feature Selection Challenge described in Guyon et al. (2004).[4] Data sets cover a large field of the feature selection problem since examples are

4. Data sets are available on http://www.nipsfsc.ecs.soton.ac.uk/datasets/.

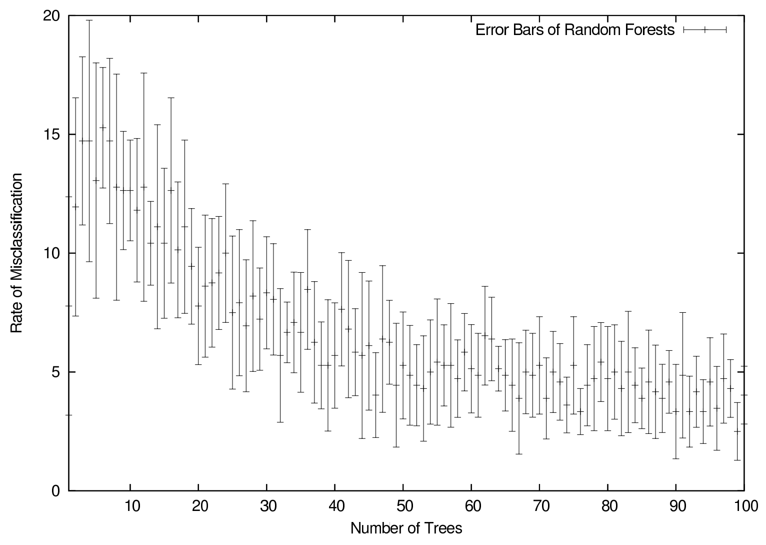


Figure 26: Error bars of Random Forests in connection with the number of trees used computed by cross-validation.

taken in several areas (Microarray data, Digit classification, Synthetic examples, Text recognition and Drug discovery). Results are provided using the Balanced Error Rate (BER) obtained on the validation set rather than the classical error rate.

We first performed a direct Optimal Feature Weighting algorithm on theses data sets without any feature preselection using a linear SVM for our base classifier $\mathbb{A}$. For four of the five data sets (DEXTER, DOROTHEA, GISETTE and MADELON) the numerical performances of the algorithm are significantly improved if a variable preselection is performed before running it. This preselection was based on the Fisher Score:

$$F_i = \frac{(\overline{x}_i^1 - \overline{x}_i)^2 + (\overline{x}_i^2 - \overline{x}_i)^2}{\frac{1}{n_1 - 1}\sum_{k=1}^{n_1}(x_{k,i}^1 - \overline{x}_i^1)^2 + \frac{1}{n_2 - 1}\sum_{k=1}^{n_2}(x_{k,i}^2 - \overline{x}_i^2)^2}.$$

Here n_1 and n_2 are the numbers of samples of the training set of classes 1 and 2, $\overline{x}_i^1, \overline{x}_i^2$ and $\overline{x}_i$ assign the mean of feature i on class 1, 2 and over the whole training set. We preselect the features with Fisher Score higher than 0.01.

We then perform our Optimal Feature Weighting algorithm with the new set of features obtained by the Fisher preselection using for $\mathbb{A}$ a support vector machine with linear kernel. Figure 27 show the decreasing evolution of the mean BER on the training set for each data sets of the feature selection challenge. One instantaneously can see that OFW is much more efficient on GISETTE or ARCENE than on other data sets since the evolution of mean BER is faster and have a larger amplitude.

For computational efficiency, our weight distribution $\mathbb{P}$ is learned using a linear SVM for the basic algorithm, $\mathbb{A}$. Once this is done, an optimal nonlinear SVM is used for the final classification (the parameters of the kernel of this final SVM being estimated via cross-validation). We select the number of features used for the classification on the validation set on the basis of a 10-fold

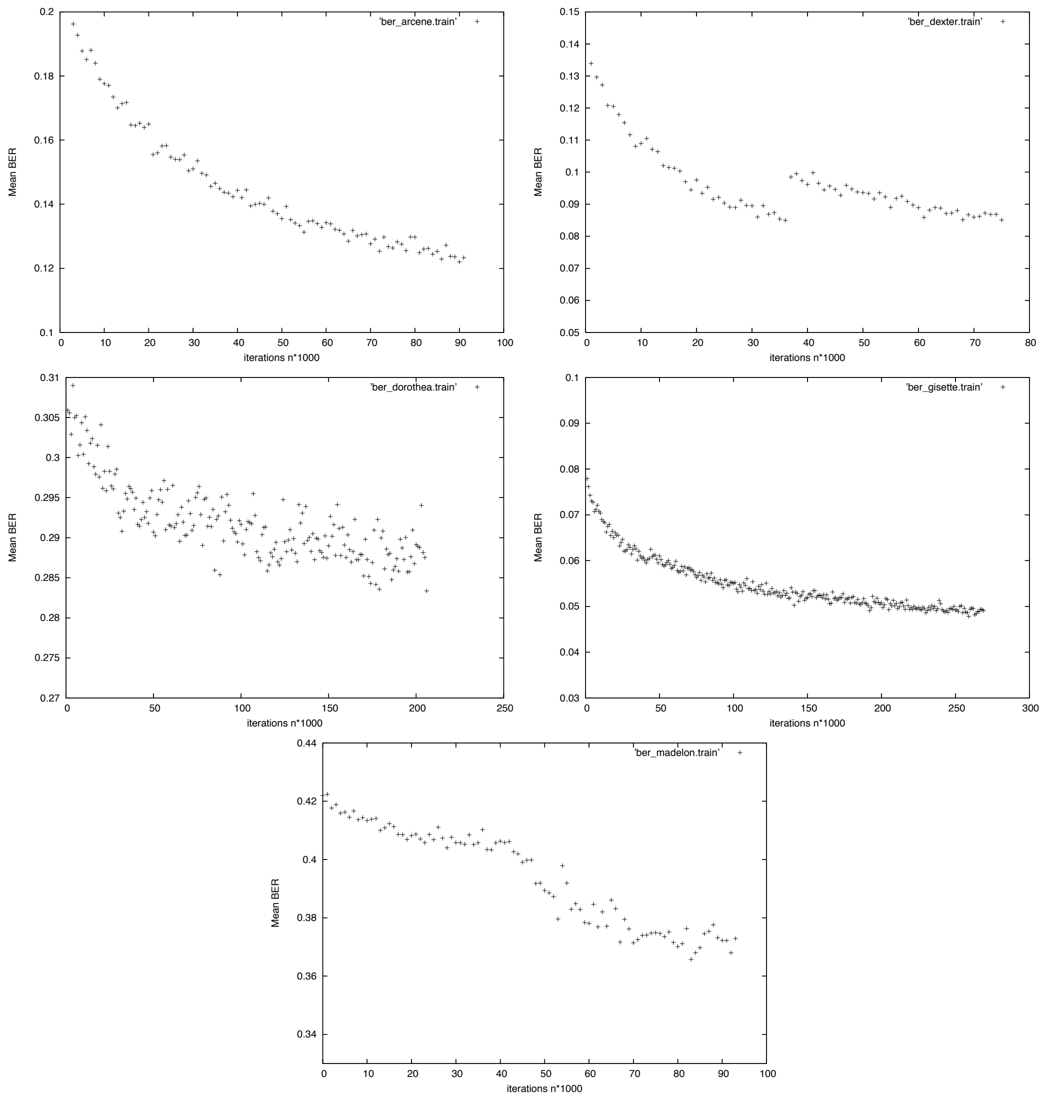


Figure 27: Evolution with iterations n of the Balanced Error Rate on the training set.

cross validation procedure on the training set. Table 2 summarizes results obtained by our OFW algorithm and, Linear SVM,[5] and others algorithms of feature selection (Transductive SVM of Wu and Li (2006), combined filter methods with svm as F+SVM of Chen and Lin (2006) and FS+SVM of Lal et al. (2006), G-flip of Gilad-Bachrach et al. (2004), Information-Based Feature Selection of Lee et al. (2006), and analysis of redundancy and relevance (FCBF) of Yu and Liu (2004)). We select these methods since they are meta algorithms (as OFW method) whose aim is to optimize the feature subset entry of standard algorithms. These results are those obtained on the Validation Set since most of the papers previously cited do not report results on the Test Set. One can show that most of these methods outperform the performance of SVM without any selection.

5. Reader can refer at http://www.nipsfsc.ecs.soton.ac.uk/results for other results obtained by feature selection procedure or different classification algorithms.

Fisher criterion is numerically effective and can exhibit very reduced sets of features, but using it alone provides performances that are below those reported in Table 2. FS+SVM and F+SVM, like all filter approaches, are equally effective and perform quite well, but requires the estimation of several thresholds, and suffer from lack of theoretical optimization background. The G-flip algorithm is to find a growing sequence of features that successively maximize the margin of the classifier. The main idea is consequently not so far from the OFW approach, even though we pick up a new feature subset at each iteration. Results are comparable with OFW and authors obtained generalization error bounds. The Transductive SVM incorporates a local optimization on the training set for a cost function related to the performances of SVM, and updates a parameter defined on each feature using coefficients of the hyperplanes constructed at each step. This approach has the drawback of high computation cost, can fail in the local optimization step and requires to tune many parameters, but obtains interesting results and suggests further developments on model selection. FCBF, which does not intend directly to increase the accuracy of any classifier as a wrapper algorithm, selects the features by identifying the redundancy between features and relevance analysis of variables. The resulting algorithms (FCBF-NBC and FCBF-C4.5) obtains very good results and is numerically simple to handle. However, this approach does not provide any theoretical measure of efficiency selection with respect to the accuracy of classification.

Our method is competitive on all data sets but DOROTHEA. The OFW algorithm is particularly good on the GISETTE data set. Moreover, we outperform most of methods based on a filter + svm approach.

Table 3 provides our results on the Test Set as well as the results of the best challenge participants. Looking at BER, best results of the challenge outperforms our OFW approach, but this comparison seems unfair since best entries are classification algorithm rather than features selection algorithm (most of features are kept to treat the data) and the difference of BER is not statistically significantly different except for the DOROTHEA database. We add moreover recent results on these 5 data sets obtained by quite simple filter methods Guyon et al. (2006) that reach remarkable BER results.

6. Discussion and Conclusion

We first start with a detailed comparison of the several results obtained during the experimental section.

6.1 Discussion

¿From the previous empirical study, we can conclude that OFW can dramatically reduce the dimension of the feature space while preserving the accuracy of classification and even enhance it in many cases. We observe likewise that we obtain results comparable to those of reference algorithms like RFE or RF. In most cases, the learning process of $\mathbb{P}_\infty$ is numerically easy to handle with our method and the results on test set are convincing. Besides the accuracy of classifier, another interesting advantage of OFW is the stability of the subsets which are selected when we run several bootstrap version of our algorithm. Further works could include numerical comparisons on the stability of several algorithms using for instance a bootstrap average of Hamming distances as it is performed in Dune et al. (2002).

Nevertheless, in some rare cases (DEXTER or DOROTHEA), learning the optimal weights is more complicated: in the case of DEXTER database, we can guess from Figure 27 that our

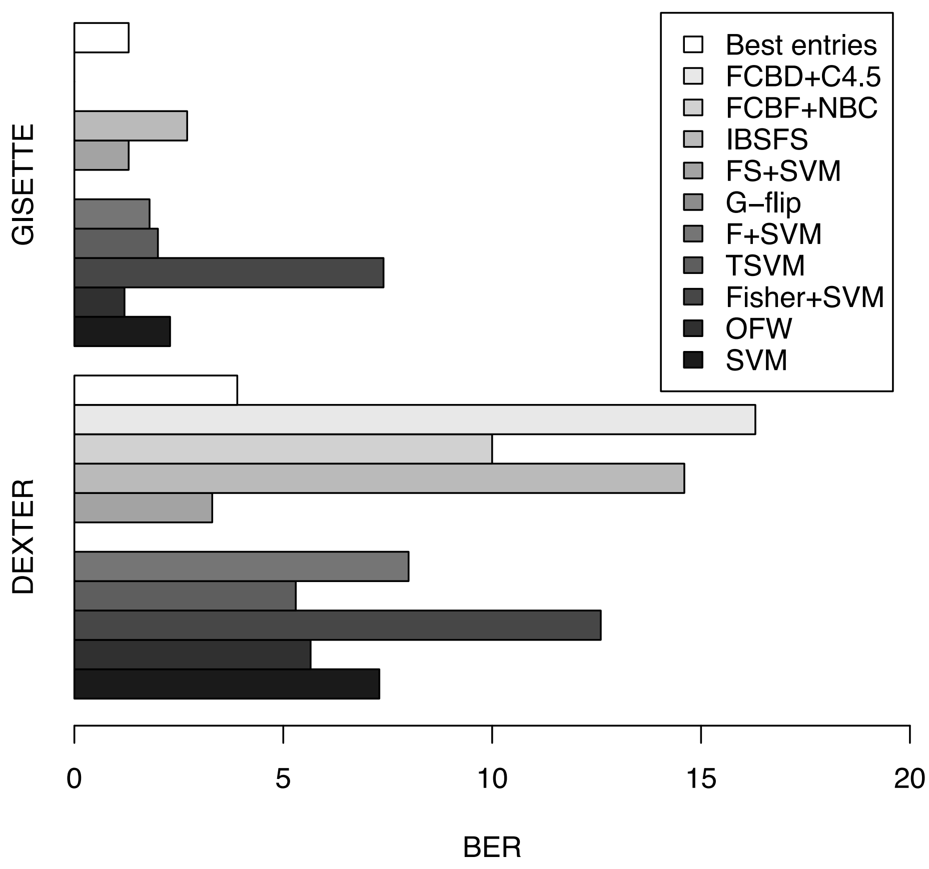


Figure 28: Performances of several algorithms on the Validation Set of the FSC. Zero bar correspond to missing values.

stochastic algorithm has been temporarily trapped in a neighborhood of a local minimum of our energy $\mathcal{E}$. Even if the OFW has succeeded in escaping the local minimum after a while, this still reduces drastically the convergence speed and the final performance of classification on the validation set. In the case of DOROTHEA, the results of SVM are quite irregular according to subsets selected along time (see Figure 27) and the final performance, as all methods based on SVM classifiers noticed in Table 2, is not as good as other reported for OFW (see best BER of the challenge in Table 3 obtained without using any SVM as final classifier). At last, results obtained by OFW are a little bit worse than those obtained by filtering techniques of Guyon et al. (2006) (see Table 3) that perform efficient feature selection. Note also that all the results obtained require larger feature subsets than OFW and use a larger amount of probes in the set of selected features. To make

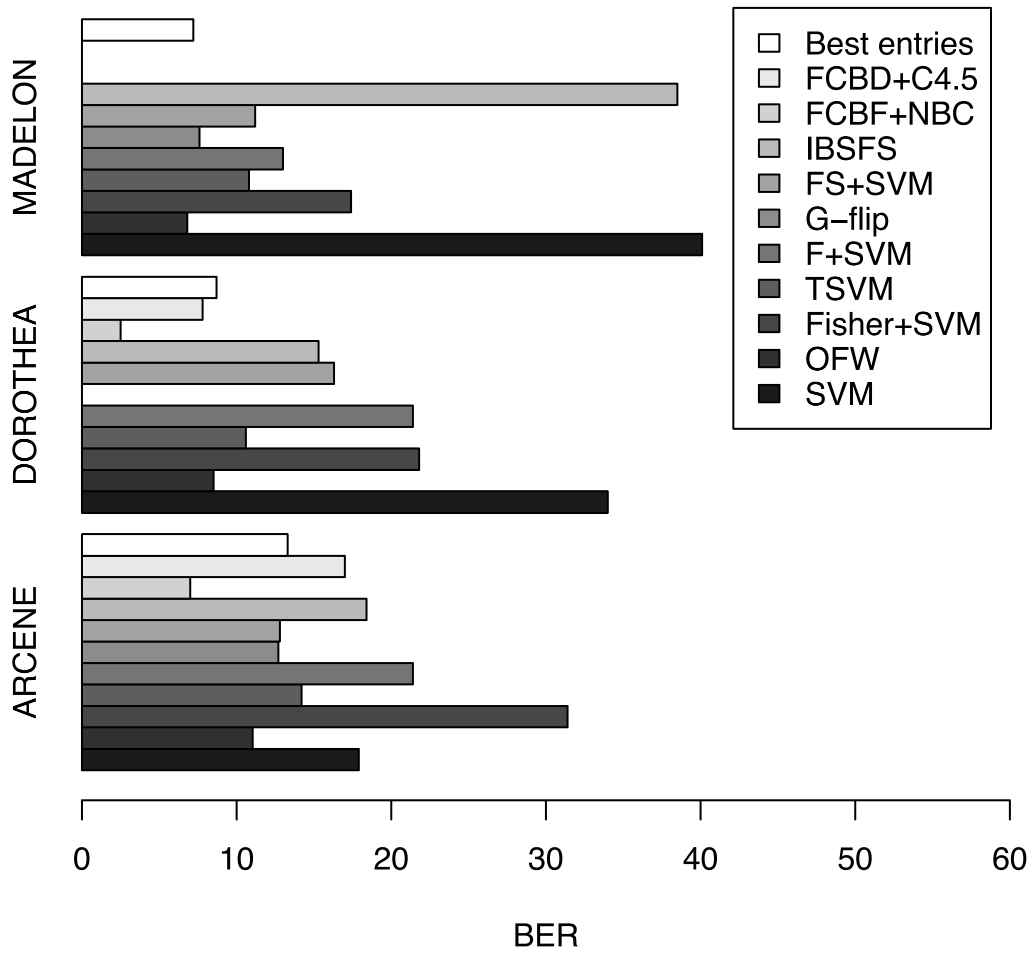


Figure 29: Performances of several algorithms on the Validation Set of the FSC. Zero bar correspond to missing values.

a comparison of their efficiency, we compute the subsets obtained by these filtering methods with the number of features used for OFW. These results are reported in the last line of Table 3. One can see that filter methods obtained poorer performances with a reduced number of features, one can note that on the DOROTHEA data set, the SVM completely miss one of the two unbalanced class and obtained bad results.

Another point of interest is the fraction of probes finally selected by each methods. Probes are artificial features added at random in each data set with statistical distributions similar to the one

Data Set	ARCENE	DEXTER	DOROTHEA	GISETTE	MADELON
BER of SVM	17.86	7.33	33.98	2.10	40.17
BER of OFW	11.04	5.67	8.53	1.1	6.83
BER of Fisher + SVM	31.42	12.63	21.84	7.38	17.4
% features selected	(3.80)	(1.43)	(0.01)	(6.54)	(2.80)
BER of TSVM	14.2	5.33	10.63	2	10.83
% features selected	(100)	(29.47)	(0.5)	(15)	(2.60)
BER of F+SVM	21.43	8	21.38	1.8	13
% features selected	(6.66)	(1.04)	(0.445)	(18.2)	(2.80)
BER of G-flip	12.66				7.61
% features selected	(0.76)				(3.60)
BER of FS+SVM	12.76	3.3	16.34	1.3	11.22
% features selected	(47)	(18.6)	(1)	(34)	(4)
BER of IBFS	18.41	14.60	15.26	2.74	38.5
% features selected	(1.85)	(5.09)	(0.77)	(9.30)	(2.40)
BER of FCBF+NBC	7	10	2.5		
% features selected	(0.24)	(0.17)	(0.5)		
BER of FCBF+C4.5	17	16.3	7.8		
% features selected	(0.24)	(0.17)	(0.5)		

Table 2: Performances of OFW and other meta algorithms on the Validation Set of the FSC, BER are given in percentage. The best results are in bold characters. The second line compares OFW with the simple Fisher scoring method with the same amount of features and show the error bars obtained by OFW.

Data Set	ARCENE	DEXTER	DOROTHEA	GISETTE	MADELON
BER of OFW	11.54	4.8	14.43	1.35	6.78
% features	(3.80)	(1.31)	(0.04)	(8.18)	(3.2)
Best Challenge BER	13.3	3.9	8.7	1.3	7.2
	(100)	(1.52)	(100)	(100)	(100)
Guyon et al. (2006)	10.48	3.25	9.3	0.98	6.22
	(14)	(22.5)	(0.7)	(15.68)	(4)
BER of Filters	14.21	4.8	41.33	4.54	7.33
	(3.80)	(1.31)	(0.04)	(8.18)	(3.2)

Table 3: Performances of OFW on the Test Set of the FSC, BER are given in percentage.

Data Set	ARCENE	DEXTER	DOROTHEA	GISETTE	MADELON
Probes of OFW	0.79	19.47	2.56	0	0
Probes of Best entry	30	12.87	50	50	96
Guyon et al. (2006)	0.36	55.38	22.14	49.23	0
Probes of Filters methods	7.63	54.58	33.33	45.97	0

Table 4: Fractions of probes selected by OFW and other algorithms on the FSC, fractions are given in percentage.

of some real features, but these probes do not carry any information on the class labels of signals. Thus, a good feature selection algorithm should obtained a small fraction of probes on the final selection. We show in Table 4 the fraction of probes obtained by the methods cited in Table 3. One can remark that OFW is particularly effective to reduce the amount of probes of any data sets (see GISETTE for instance).

Better results of OFW have been obtained for two special cases of databases which are microarray data (ARCENE and Leukemia) and image recognition data (USPS, GISETTE and Faces). SVM initially performs well on this data sets, but OFW significantly improve the performance without any selection.

More generally, OFW seems to behave well and to boost accuracy of algorithms $\mathbb{A}$ that have initial performance that vary smoothly with respect to small changes in the feature set. Even if our learning procedure is computed in a very large dimensional training set, recent work have shown that in the context of classification, bootstrap approaches (as it is done in OFW) do not introduce a supplementary important bias (Singhi and Liu, 2006) and it is equally what we can conclude in our case.

In the first synthetic example, one can make the important remark that reusable features are mainly favored by OFW. The algorithm classes those which are relevant, but not reusable in a second group. This point looks favorable to our model of ”frequency of use”.

Our approach does not address the issue of redundancy (two similar features would most likely receive the same weight). Some ideas in how to take this into account are sketched in the concluding section.

6.2 Computational Considerations

We have performed our experiments using a C++ compiler with a 2.2 GHz 1 Go RAM processor pentium IV PC on a Debian system. The learning time of OFW mostly depends on the initial number of variables in the feature space and the step of our stochastic scheme; for the Leukemia database which contains 3859 genes, learning took about one hour.

However, OFW can be easily implemented with parallel techniques since at each step of the stochastic procedure, one can test several subsets of the feature set still using the same update formula of $\mathbb{P}_n$. Moreover, we remark that it can be effective for the calculation time to first filter out very irrelevant features (selected for instance by a Fisher Score) and run the OFW procedure.

Another option would be to use algorithm $\mathbb{A}$ simpler than SVM, CART or NN, based on basic statistical tools as likelihood or mutual information whilst performing a final decision with SVM for instance.

6.3 Conclusion, Future Work

Our approach introduces a mathematical model to formalize the search for optimal features. Our selection of features is done by learning a probability distribution on the original feature set, based on a gradient descent of an energy $\mathcal{E}$ within the simplex $\mathcal{S}_{\mathcal{F}}$.

The numerical results show that the performance is significantly improved over an initial rule in which features are simply uniformly distributed. Our Optimal Feature Weighting method is moreover competitive in comparison with other feature selection algorithms and leads to an algorithm which does not depend on the nature of the classifier $\mathbb{A}$ which is used, whereas, for instance, RFE or L0-SVM are only based on SVM.

Our future work will enable the feature space $\mathcal{F}$ to be also modified during the algorithm, by allowing for combination rules, creation and deletion of tests, involving a hybrid evolution in the set of probability measures and in the feature space. This will be implemented as a generalization of our constrained diffusion algorithm to include jumps in the underlying feature space.

Another development would be to speed up the learning procedure using stochastic algorithm techniques to handle even larger databases without using a combination of filter and wrapper method.

Acknowledgments

We thank the anonymous referees and the action editor for their helpful comments that permit to significantly complete and improve both the experimental section and the general clarity of this manuscript.

References

Y. Amit and D. Geman. A computational model for visual selection. *Neural Computation*, 11(7): 1691 – 1715, 1999.

Y. Amit and D. Geman. Shape quantization and recognition with randomized trees. *Neural Computation*, 9(7):1545–1588, 1997.

M. Benaïm. Convergence with probability one of stochastic approximation algorithms whose average is cooperative. *Nonlinearity*, 13(3):601–616, 2000.

M. Benaim. A dynamical system approach to stochastic approximations. *SIAM Journal on Control and Optimization*, 34(2):437–472, 1996.

A. Benveniste, M. Métivier, and P. Priouret. *Adaptive Algorithms and Stochastic Approximations*, volume 22 of *Applications of Mathematics (New York)*. Springer-Verlag, Berlin, 1990.

J. Bins and B. A. Draper. Feature selection from huge feature sets. In *Proceedings of the Eighth International Conference On Computer Vision*, volume 2, pages 159–165, 2001.

A. Blum and R. L. Rivest. Training a 3-node neural network is NP-complete. *Neural Networks*, 5 (1):117– 127, 1992.

L. Breiman. Arcing classifiers. *The Annals of Statistics*, 26(3):801–849, 1998.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

R. Buche and H. J. Kushner. Rate of convergence for constrained stochastic approximation algorithms. *SIAM Journal on Control Optimization*, 40(4):1011–1041, 2001.

O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1):131–159, 2002.

Y. W. Chen and C. J. Lin. Combining SVMs with various feature selection strategies. In *Feature extraction, foundations and applications*. Springer-Verlag, Berlin, 2006.

S. Cohen, E. Ruppin, and G. Dror. Feature selection based on the Shapley value. In *International Joint Conference on Artificial Intelligence*, pages 665–670, 2005.

T. Cover and J. Thomas. *Information Theory*. Wiley, New York, 1991.

K. Deb and R. Reddy. Reliable classification of two-class cancer data using evolutionary algorithms. *BioSystems*, 72(1):111 – 129, 2003.

T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

C.L. Blake D.J. Newman, S. Hettich and C.J. Merz. UCI repository of machine learning databases, 1998. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

M. Duflo. *Algorithmes Stochastiques*, volume 23 of *Mathématiques & Applications*. Springer-Verlag, Berlin, 1996.

K. Dune, P. Cunningham, and F. Azuaje. Solutions to instability problems with sequential wrapper-based approaches to feature selection. In *Technical Report-2002-28*, Department of Computer Science Courses, Trinity College, Dublin, 2002.

P. Dupuis and H. Ishii. On Lipschitz continuity of the solution mapping to the Skorokhod problem, with applications. *Stochastics and Stochastics Reports*, 35(1):31–62, 1991.

P. Dupuis and K. Ramanan. Convex duality and the Skorokhod problem. I, II. *Probability Theory and Related Fields*, 115(2):153–195, 197–236, 1999.

F. Fleuret. Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research*, 5:1531–1555, 2004.

F. Fleuret and D. Geman. Coarse-to-fine face detection. *International Journal of Computer Vision*, 41(1-2):85–107, 2001.

J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 28(2):337–407, 2000.

S. Gadat. Apprentissage d'un vocabulaire symbolique pour la détection d'objets dans une image. *PhD thesis, École Normale Supérieure de Cachan, France*, 2004.

S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58, 1992.

R. Gilad-Bachrach, A. Navot, and N. Tishby. Margin based feature selection - theory and algorithms. In *Proceedings of the 21rd International Conference on Machine Learning*, 2004.

T.R. Golub, D.K. Slonim, P. Tamazyo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, and E.S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286: 531– 537, 1999.

I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.

I. Guyon, S. Gunn, A. B. Hur, and G. Dror. Result analysis of the nips 2003 feature selection challenge. In *Proceedings of the Neural Information Processing Systems*, pages 545 – 552, 2004.

I. Guyon, J. Li, T. Mader, P. Pletscher, G. Schneider, and M. Uhr. Teaching machine learning from examples. *Unpublished*, 2006.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning : Data Mining, Inference, and Prediction*. Springer-Verlag, Berlin, 2001.

T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining*, 2002.

T. Joachims and R. Klinkenberg. Detecting concept drift with support vector machines. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 487–494, 2000.

C. Jutten and J. Hérault. Blind separation of sources, part 1: An adaptative algorithm bases on neuromimetic architecture. *Signal Processing*, 24(1):1–10, 1991.

H. J. Kushner and G. G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications*, volume 35 of *Applications of Mathematics*. Springer-Verlag, New York, second edition, 2003.

T. N. Lal, O. Chapelle, and B. Schölkopf. Combining a filter method with svms. In *Feature extraction, foundations and applications*. Springer-Verlag, 2006.

S. K. Lee, S. J. Yi, and B. T. Zhang. Combining information-based supervised and unsupervised feature selection. In *Feature extraction, foundations and Applications*. Springer-Verlag, 2006.

Y. Liu and J. R. Render. Video retrieval under sparse training data. In *Proceedings of the Second International Conference on Image and Video Retrieval*, pages 406–413, 2003.

D.J.C. MacKay. A practical Bayesian framework for back propagation networks. *Neural Computation*, 3:448–472, 1992.

B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, 2002.

B. Schölkopf, C. Burges, and V. Vapnik. Extracting support data for a given task. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pages 252–257, 1995.

P. Simard and Y. LeCun. Memory based character recognition using a transformation invariant metric. In *Proceedings of the 12th International Conference on Pattern Recognition*, pages 262–267, 1998.

S. Singhi and H. Liu. Feature subset selection bias for classification learning. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 849 – 856, 2006.

Y. Sun and J. Li. Iterative relief for feature weighting. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 913 – 920, 2006.

V. Vapnik. *The Nature of Statistical Learning Theory*. Statistics for Engineering and Information Science. Springer-Verlag, New York, second edition, 2000.

V. Vapnik. *Statistical Learning Theory*. Adaptive and Learning Systems for Signal Processing, Communications, and Control. John Wiley & Sons Inc., New York, 1998.

J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In *Neural Information Processing Systems*, pages 668–674, 2000.

J. Weston, A. Elisseeff, B. Schlkopf, and M. E. Tipping. Use of the zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

Z. Wu and C. Li. Feature selection for classification using transductive support vector machines. In *Feature Extraction, Foundations and Applications*. Springer-Verlag, Berlin, 2006.

E. Xing, M. I. Jordan, and S. Russel. Feature selection for high-dimensional genomic microarray data. In *Proceedings of the Eighteenth Internation Conference on Machine Learning*, pages 601–608, 2001.

L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004.

Journal of Machine Learning Research 8 (2007) 549-587 Submitted 12/05; Revised 8/06; Published 3/07

Learning Horn Expressions with LOGAN-H

Marta Arias MARTA@CS.COLUMBIA.EDU
Center for Computational Learning Systems
Columbia University
New York, NY 10115, USA

Roni Khardon RONI@CS.TUFTS.EDU
Department of Computer Science
Tufts University
Medford, MA 02155, USA

Jérôme Maloberti MALOBERT@LRI.FR
Laboratoire de Recherche en Informatique
Université Paris-Sud
F-91405, Orsay, France

Editor: Stefan Wrobel

Abstract

The paper introduces LOGAN-H —a system for learning first-order function-free Horn expressions from interpretations. The system is based on an algorithm that learns by asking questions and that was proved correct in previous work. The current paper shows how the algorithm can be implemented in a practical system, and introduces a new algorithm based on it that avoids interaction and learns from examples only. The LOGAN-H system implements these algorithms and adds several facilities and optimizations that allow efficient applications in a wide range of problems. As one of the important ingredients, the system includes several fast procedures for solving the subsumption problem, an NP-complete problem that needs to be solved many times during the learning process. We describe qualitative and quantitative experiments in several domains. The experiments demonstrate that the system can deal with varied problems, large amounts of data, and that it achieves good classification accuracy.

Keywords: inductive logic programming, subsumption, bottom-up learning, learning with queries

1. Introduction

The field of Inductive Logic Programming (ILP) deals with the theory and the practice of generating first order logic rules from data. ILP has established a core set of methods and systems that have proved useful in a variety of applications (Muggleton and DeRaedt, 1994; Bratko and Muggleton, 1995). As in much of the work on propositional rule learning, ILP systems that learn rules can be divided into *bottom up* methods and *top down* methods. The latter typically start with an empty rule and grow the rule condition by adding one proposition at a time. Bottom up methods start with a "most specific" rule and iteratively generalize it, for example by dropping propositions from the condition. There are examples of *bottom up* systems in the early days of ILP: for example, the GOLEM system (Muggleton and Feng, 1992) used relative least general generalization (Plotkin, 1970, 1971) within a bottom up search to find a hypothesis consistent with the data. Related generalization operators were also used by Muggleton and Buntine (1992). On the other hand, much of the research following this (Quinlan, 1990; Muggleton, 1995; De Raedt and Van Laer, 1995; Blockeel and De Raedt, 1998) used top down search methods to find useful hypotheses.[1] This paper introduces the system LOGAN-H (Logical Analysis for Horn expressions) that implements a bottom up learning algorithm.

The system comes in two main modes. In *batch mode* the system performs the standard supervised learning task, taking a set of labeled examples as input and returning a hypothesis. In *interactive mode* the system learns by asking questions. The questions are the Equivalence Queries and Membership Queries (Angluin, 1988) that have been widely studied in computational learning theory, and in particular also in the context of ILP (Arimura, 1997; Reddy and Tadepalli, 1998; Krishna Rao and Sattar, 1998; Khardon, 1999b,a; Arias and Khardon, 2002).

1.1 A Motivating Example: Learning from Graphs

We introduce the problem and some of the algorithmic ideas through a simple example in the context of learning from labeled graphs where both nodes and edges may have labels. In fact this is one of the applications of learning from interpretations where the atom-bond relations of a molecule can be seen as such a graph and this framework has proved useful for predicting certain properties of molecules.

In the following, node labels are marked by $n_1, n_2, \ldots$, edge labels are $e_1, e_2, \ldots$ and we apply them to nodes or pairs of nodes appropriately. Consider a class of graphs that is characterized by the following rules:

$$\begin{aligned} R_1 &= \forall x_1, x_2, x_3, x_4, \\ &\quad e_1(x_1,x_2), e_1(x_2,x_3), e_2(x_3,x_4), e_2(x_4,x_1) \rightarrow e_1(x_1,x_3), \\ R_2 &= \forall x_1, x_2, x_3, x_4, \\ &\quad n_1(x_1), n_1(x_2), n_2(x_3), n_3(x_4), e_1(x_1,x_2), e_1(x_1,x_3), e_1(x_1,x_4) \rightarrow e_2(x_2,x_3). \end{aligned}$$

Both rules imply the existence of an edge with a particular label if some subgraph exists in the graph. Consider the following graphs, also illustrated in Figure 1, that may be part of the data:

$(g_1) = e_1(1,2), e_1(2,3), e_2(3,4), e_2(4,1), e_1(1,3), e_1(3,5), e_2(4,5).$
$(g_2) = e_1(1,2), e_1(2,3), e_1(3,4).$
$(g_3) = e_1(1,2), e_1(2,3), e_2(3,4), e_2(4,1), e_1(2,5), e_1(5,6), e_1(6,3), e_2(1,2), e_1(2,4).$
$(g_4) = e_1(1,2), e_1(2,3), e_2(3,4), e_2(4,1), e_1(3,5), e_1(4,5), e_2(1,2), e_2(2,3).$
$(g_5) = n_1(1), n_1(2), n_2(3), n_3(4), e_1(1,2), e_1(1,3), e_1(1,4), e_1(2,5), e_1(3,6), e_1(5,6), e_1(6,4).$

It is easy to see that g_1 and g_2 satisfy the constraints given by the rules. We call such graphs positive examples. The graphs g_3 and g_4 violate the first rule and g_5 violates the second rule. We call such graphs negative examples. Now given a data set of such graphs how can we go about inferring the underlying rules? In the following we ignore the question of identifying the conclusions of rules and illustrate some of the basic ideas and steps used in our algorithm.

1. Some exceptions exist. STILL (Sebag and Rouveirol, 2000) uses a disjunctive version space approach which means that it has clauses based on examples but it does not generalize them explicitly. The system of Bianchetti et al. (2002) uses bottom up search with some ad hoc heuristics to solve the challenge problems of Giordana et al. (2003).

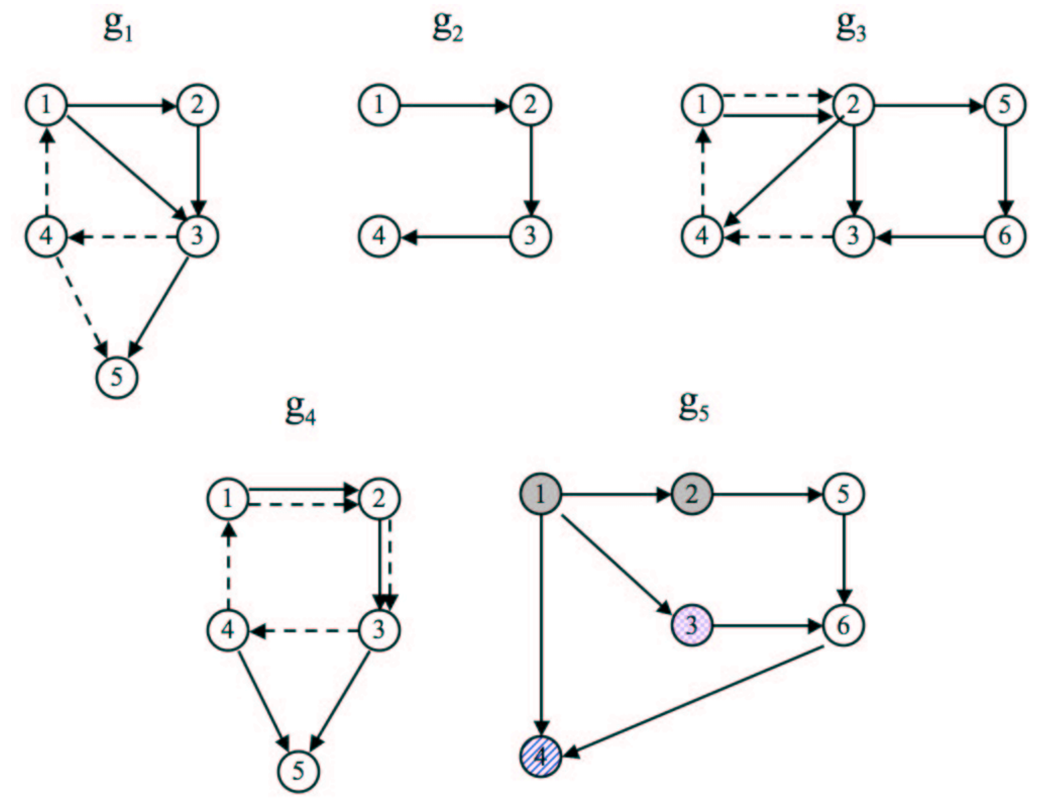


Figure 1: Representation of graphs g_1 to g_5. Solid lines represent $e_1(\cdot,\cdot)$ edges, dotted lines represent $e_2(\cdot,\cdot)$ edges. Solidly filled nodes represent $n_1(\cdot)$ nodes, checked nodes represent $n_2(\cdot)$ nodes, and nodes filled with diagonal lines represent $n_3(\cdot)$ nodes.

Consider taking one negative example, say g_3 and trying to extract information from it. If we could discover that only 4 nodes are required to "make it negative" and these nodes are $1,2,3,4$ then we could get a smaller graph to work with and use that as a pattern for our rule condition. In particular projecting g_3 onto the nodes $1,2,3,4$ gives

$$(g_{3_{min}}) = e_1(1,2), e_1(2,3), e_2(3,4), e_2(4,1), e_2(1,2), e_1(2,4).$$

To discover this "minimized" version of g_3 we can drop one node from the graph and then try to find out whether that node was irrelevant, that is, whether the resulting pattern is still a negative graph. If we are allowed to ask questions then we can do this directly. Otherwise, we use a heuristic to evaluate this question using the data set. Iterating this procedure of dropping an object gives us a minimized negative graph. Below, we call this the minimization procedure.

Now consider taking two examples, say g_3 and g_4 and trying to extract information from them. Again the structure of examples has important implications. If we could discover that only 4 nodes are required for both examples, and that in both cases these are nodes $1,2,3,4$, we can focus on the shared structure in the two graphs. In this case we get

$$(g_{3,4}) = e_1(1,2), e_1(2,3), e_2(3,4), e_2(4,1), e_2(1,2).$$

so the pattern is even closer to the rule than the one in $g_{3_{min}}$. Notice that the numbering of the nodes violating the rule in the two graphs will not always coincide. Therefore, in general renaming or "aligning" of the violating nodes will be necessary and our algorithm must search for such an alignment. Again we must verify that the resulting pattern still captures one of our rules. For example if we tried this operation using g_3 and g_5 we will not find any common relevant core since they violate different rules and do not have a shared rule structure. However, if we succeed in finding a common structure then we can make substantial progress toward finding good rules. Below we call this the pairing procedure since it pairs and aligns two examples.

Our algorithm uses minimization and pairing as well as some other ideas in a way that guarantees finding a consistent hypothesis. Bounds for the hypothesis size and the complexity of the algorithm are given by Khardon (1999b) for the case that the algorithm can ask questions and get correct answers for them. In the case answers to the questions are estimated from the data we can provide similar bounds under somewhat strong assumptions on the data set. As our experiments show, the algorithm performs well on a range of problems.

1.2 General Properties of the System

Some of the properties of LOGAN-H were illustrated above. The system learns in the model of *learning from interpretations* (De Raedt and Dzeroski, 1994; De Raedt and Van Laer, 1995; Blockeel and De Raedt, 1998) where each example is an interpretation, also called a first order structure in logic terminology (Chang and Keisler, 1990). Roughly, an interpretation is the description of some scenario. Interpretations have a domain which consists of a set of objects, and a list of relations between objects in its domain that are "true" or hold in the particular scenario they describe. The system learns function free Horn rules meaning that it learns over relational data but all arguments of predicates are universally quantified variables. We do not allow for function symbols or constants in the rules.

The system solves the so-called multi-predicate learning problem, that is, it can learn multiple rules and different rules can have different conclusions. The hypothesis of the system may include recursive clauses where the same predicate appears both in the condition of the rule and in the conclusion (obviously with different arguments). In the previous example, R_1 is a recursive rule but R_2 is not.

In contrast with most systems, LOGAN-H learns all the rules simultaneously rather than learning one rule at a time (cf. Bratko, 1999). The system uses bottom up generalization in this process. In terms of rule refinement, the system can be seen as performing large refinement steps. That is, a rule structure is changed in large chunks rather than one proposition at a time. Again this is in contrast with many approaches that try to take minimal refinements of rules in the process of learning.

As mentioned above the system can run in two modes: *interactive* where the algorithm asks questions in the process of learning, and *batch* where standard supervised learning is performed. The learning algorithms in both modes are based on the algorithm of Khardon (1999b) where it was shown that function-free Horn expressions are learnable from equivalence and membership queries. The interactive mode algorithm essentially uses this algorithm but adds several features to make the system more efficient. The batch mode algorithm uses the idea of simulation and can be thought of as trying to answer the queries of the interactive algorithm by appealing to the data set it is given as input.

Efficiency considerations are important in any ILP system and in particular in bottom up learning, since these systems tend to start with larger and more complex clauses thus requiring more resources in the beginning. The system includes several techniques that speed up run time. Perhaps the most important is the use of fast implementations for subsumption, an NP-Complete matching problem that needs to be solved many times in our algorithm. The paper introduces two new methods for this problem, in addition to using the DJANGO algorithm (Maloberti and Sebag, 2004) which is based on constraint satisfaction techniques. One of the new methods modifies DJANGO to use a different representation of the subsumption problem and requires slightly different constraint satisfaction algorithms. The other method manipulates partial substitutions as tables and iteratively applies joins to the tables to find a solution.

The system incorporates further useful facilities. These include a module for pruning rules, a standard technique that may improve accuracy of learned rules. Another module is used for discretizing real valued arguments. This increases the applicability since many data sets have some numerical aspects. Both of these use known ideas but they raise interesting issues that have not been discussed in the literature, mainly since they are relevant for bottom up learners but not for top down learners.

1.3 Experimental Results

We have experimented with both modes of the system and challenging learning problems. The paper reports on qualitative experiments illustrating the performance of the system on list manipulation procedures (15-clause program including standard procedures) and a toy grammar learning problem. The grammar problem introduces the use of background information to qualify grammar rules that may be of independent interest for other settings.

We also describe quantitative experiments with several ILP benchmarks. In particular, the performance of the system is demonstrated and compared to other systems in three domains: the Bongard domain (De Raedt and Van Laer, 1995), the KRK-illegal domain (Quinlan, 1990), and the Mutagenesis domain (Srinivasan et al., 1994). The results show that our system is competitive with previous approaches, both in terms of run time and classification accuracy, while applying a completely different algorithmic approach. This suggests that bottom up approaches can indeed be used in large applications.

The experiments also demonstrate that the different subsumption algorithms are effective on a range of problems but no one dominates the others with LOGAN-H. Further experiments evaluating the subsumption methods on their own show that our modified DJANGO method is a promising approach when hard subsumption problems need to be solved.

1.4 Summary and Organization

The main contribution of the paper is introducing the LOGAN-H system with its learning algorithms and implementation heuristics. The system is a bottom up ILP system using ideas from learning theory in its algorithms. The paper demonstrates through various experiments that the system and its algorithms provide an interesting alternative to top down learning which is the common approach in ILP. The paper also makes a contribution to the study of efficient subsumption algorithms. We develop some new subsumption algorithms and evaluate them both in the context of machine learning and independently. One of the new methods, called DJANGOPRIMAL below, seems particularly promising when hard subsumption problems need to be solved.

The paper is organized as follows. Section 2 formalizes the learning setting and problem. Section 3 describes our learning algorithm and its different variants. Section 4 describes three subsumption engines that our system uses. Section 5 describes extensive experiments that demonstrate the validity of our method and of the subsumption procedures. Finally, in the appendix the reader can find multiple details on several implementation improvements and heuristics used to optimize the system.

2. Notation and Problem Settings

We start with some basic notation from logic and logic programming as used in the paper. For a general introduction to these topics see Chang and Keisler (1990) and Lloyd (1987). We assume that the learner is given its "vocabulary" in advance. Formally this is a fixed *signature*, a set of predicates each with its associated arity. The expressions we consider do not allow any constants or other function symbols so only variables can be used as arguments to predicates. An atom is a predicate with an appropriate list of arguments. A Horn clause (sometimes called a Horn rule or just a rule) is an expression $C = (\wedge_{n \in \mathrm{N}} n) \rightarrow P$, where N is a set of atoms and P is an atom. In the examples that follow we assume a signature with two predicates $p()$ and $q()$ both of arity 2. For example, $c_1 = \forall x_1, \forall x_2, \forall x_3, [p(x_1,x_2) \wedge p(x_2,x_3) \rightarrow p(x_1,x_3)]$ is a clause in the language. In this paper all variables in the clause are universally quantified and we often omit the quantifiers in expressions. The conjunction on the left of the implication is called the condition (also antecedent) of the rule and P is called the conclusion (also consequent). Clauses may have empty consequents. A universally quantified function-free Horn expression is a conjunction of Horn clauses. The goal of our learning algorithms is to learn such expressions.

An expression is given a truth value relative to an interpretation of the symbols in the signature. An *interpretation* lists a domain of elements and the truth values of predicates over them. For example, the interpretation

$$e_1 = ([1,2,3],[p(1,2),p(2,3),p(3,1),q(1,3)])$$

has domain $[1,2,3]$; by convention, the four atoms listed are true in the interpretation and other atoms are false. The *size* of an interpretation is the number of atoms true in it, so that $\mathit{size}(e_1) = 4$. The interpretation e_1 *falsifies* (we sometimes also say *violates*) the clause above. To see this, substitute $\{1/x_1, 2/x_2, 3/x_3\}$). On the other hand

$$e_2 = ([a,b,c,d],[p(a,b),p(b,c),p(a,c),p(a,d),q(a,c)])$$

satisfies the clause. We use standard notation $e_1 \not\models c_1$ and $e_2 \models c_1$ for these facts.

Throughout the paper we use the following notation. The *clause set* $[s,c]$ (where $c \neq \emptyset$) represents the conjunction of clauses $\bigwedge_{b \in c}(s \rightarrow b)$. If c is empty then no clause is represented and $[s,\emptyset]$ is in this sense "illegal". As mentioned above, Horn clauses in general do allow for "empty consequents". For example, a disjunction $(\overline{A} \vee \overline{B})$ is often described as $(A \wedge B \rightarrow \Box)$. Our system can handle such consequents but we do so by explicitly representing a predicate `false` with arity 0 which is false in all interpretations. Thus, the clause above will be represented in our system as $[[A,B],[\mathtt{false}]]$.

2.1 The Model: Learning from Interpretations

Learning from interpretations has seen growing interest in recent years (De Raedt and Dzeroski, 1994; De Raedt and Van Laer, 1995; Blockeel and De Raedt, 1998; Blockeel et al., 1999; Stolle et al., 2005). Unlike the standard ILP setting, where examples are atoms, examples in this framework are interpretations. As mentioned above we consider two settings for the learner.

The *batch learning algorithm* performs the standard supervised learning task: given a set of positive and negative examples it produces a Horn expression as its output. In this paper we measure algorithm performance by testing the hypothesis on an unseen test set, but we also mention theoretical guarantees for the algorithms implied by previous work.

The *interactive learning algorithm* requires an interface capturing Angluin's (1988) model of learning from equivalence queries (EQ) and membership queries (MQ). In this model we assume that a *target expression*—the true expression classifying the examples—exists and denote it by T. An EQ asks about the correctness of the current hypothesis. With an EQ the learner presents a hypothesis H (a Horn expression) and, in case H is not correct, receives a counter-example, positive for T and negative for H or vice versa. With a MQ, the learner presents an example (an interpretation) and is told in reply whether it is positive or negative for T. The learner asks such queries until H is equivalent to T and the answer to EQ is "yes".

3. The Learning Algorithms

We first describe the main procedures used by the learning algorithms and then describe the interactive and batch algorithms.

3.1 Basic Operations

Candidate clauses: For an interpretation I, the "relational candidates" (Khardon, 1999b) $\textit{rel-cands}(I)$ is the set of potential clauses all sharing the same antecedent and violated by I. The set $\textit{rel-cands}(I)$ is calculated in two steps. The first step results in a set of clauses $[s,c]$ such that s (the antecedent) is the conjunction of all atoms true in I and c (the conclusions) is the set of all atoms (over the domain of I) which are false in I. This clause set is such that all arguments to the predicates are domain elements of I. For example when applied to the example $e_3 = ([1,2],[p(1,2),p(2,2),q(2,1)])$ this gives

$$[s,c] = [[p(1,2),p(2,2),q(2,1)],[p(1,1),p(2,1),q(1,1),q(1,2),q(2,2)]].$$

While domain elements are not constants in the language and we do not allow constants in the rules we slightly abuse normal terminology and call this intermediate form a *ground clause set*. We then replace each domain element with a distinct variable to get

$$\textit{rel-cands}(I) = \textit{variabilize}([s,c]) = [s',c']$$

where $\textit{variabilize}()$ replaces every domain elements with a distinct variable. For example, the clause set $\textit{rel-cands}(e_3)$ includes among others the clauses $[p(x_1,x_2) \wedge p(x_2,x_2) \wedge q(x_2,x_1) \rightarrow p(x_2,x_1)]$, and $[p(x_1,x_2) \wedge p(x_2,x_2) \wedge q(x_2,x_1) \rightarrow q(x_1,x_1)]$, where all variables are universally quantified.

Note that there is a one to one correspondence between a ground clause set $[s,c]$ and its variabilized version. In the following we often use $[s,c]$ with the implicit understanding that the appropriate version is used.

Dropping objects: This operation can be applied to an interpretation or a clause set. When dropping an object (domain element) from an interpretation we remove the element from the domain and all atoms referring to it from the extensions of predicates. Thus if we remove object 2 from $e_1 = ([1,2,3],[p(1,2),p(2,3),p(3,1),q(1,3)])$ we get $e'_1 = ([1,3],[p(3,1),q(1,3)])$. When removing an object from a clause set $[s,c]$ we remove all atoms referring to it from s and c. For example when dropping object 1 from the clause

$$[s,c] = [[p(1,2),p(2,2),q(2,1)],[p(1,1),p(2,1),q(1,1),q(1,2),q(2,2)]]$$

we get $[s',c'] = [[p(2,2)],[q(2,2)]]$.

Minimization for Interactive Algorithm: Given a negative example I the algorithm iterates over domain elements. In each iteration it drops a domain element and asks a MQ to get the label of the resulting interpretation. If the label is negative the algorithm continues with the smaller example; otherwise it retains the previous example. The minimization ensures that the example is still negative and the domain of the example is not unnecessarily large. We refer to this as *minimize-objects*(I).

Clearly, the order by which objects are considered for removing can affect the resulting hypothesis. The theoretical guarantees for the interactive algorithm discussed below hold regardless of the order but the performance of the batch algorithm can be affected dramatically. The current system does not try to optimize this and simply drops objects in the order they are encountered in example descriptions.

Removing Wrong Conclusions: Consider a clause set $[s,c]$ in the hypothesis, say as initially generated by *rel-cands*, and consider a conclusion $p \in c$ that is wrong for s. We can identify such a conclusion if we see a positive example I such that $I \not\models [s \rightarrow p]$. Notice that since I is positive it does not violate any correct rule and since it does violate $[s \rightarrow p]$ this rule must be wrong. In such a case we can remove p from c to get a better clause set $[s, c \setminus p]$. In this way we can eventually remove all wrong conclusions and retain only correct ones.

Pairing: The pairing operation combines two clause sets $[s_a,c_a]$ and $[s_b,c_b]$ to create a new clause set $[s_p,c_p]$. It is closely related to the LGG operation of Plotkin (1970) but it avoids the exponential growth in size of the resulting clause after several applications of the operation. When pairing, we use an injective mapping from the smaller domain to the larger one. The system first pairs the antecedents by taking the intersection under the injective mapping to produce a new antecedent J. The resulting clause set is $[s_p,c_p] = [J,(c_a \cap c_b) \cup (s_a \setminus J)]$. To illustrate this, the following example shows the two original clauses, a mapping and the resulting values of J and $[s_p,c_p]$.

- $[s_a,c_a] = [[p(1,2),p(2,3),p(3,1),q(1,3)],[p(2,2),q(3,1)]]$.
- $[s_b,c_b] = [[p(a,b),p(b,c),p(a,c),p(a,d),q(a,c)],[q(c,a)]]$.
- The mapping $\{1/a,2/b,3/c\}$.
- $J = [p(1,2),p(2,3),q(1,3)]$.
- $[s_p,c_p] = [[p(1,2),p(2,3),q(1,3)],[q(3,1),p(3,1)]$.

Note that the pairing operation is not symmetric. The clause set $[s_p, c_p]$ obtained by the pairing can be more general than the original clause sets $[s_a, c_a]$ and $[s_b, c_b]$ since s_p is contained in both s_a and s_b (under the injective mapping) and it thus subsumes both s_a and s_b. Hence, the pairing operation can be intuitively viewed as a generalization of both participating clause sets. However, since we modify the consequent, by dropping some atoms and adding other atoms (from $s_a \setminus J$), this is not a pure generalization operation.

Note that since a pairing uses an injective mapping of objects we can use domain element names from either of the interpretations in the pairing. While this does not change the meaning of the clause sets this fact is used in the system to improve efficiency. This point is discussed further below.

The reasoning behind the definition of the consequent in the pairing operation is as follows. Consider a clause set $[s, c \setminus p]$ where a wrong conclusion p has already been removed. Now when we pair this clause set with another one the antecedent will be a subset of s and surely p will still be wrong. So we do not want to reintroduce such conclusions after pairing. The atoms that are removed from s while pairing have not yet been tested as conclusions for s and they are therefore added as potentially correct conclusions in the result.

3.2 The Interactive Algorithm

The interactive algorithm is basically the algorithm *A2* from Khardon (1999b). The algorithm is summarized in Table 1 where T denotes the target expression. Intuitively, the algorithm generates clauses from examples by using *rel-cands*(). It then uses dropping of domain elements and pairing in order to get rid of irrelevant parts of these clauses.

The algorithm maintains a sequence S of ground clause sets and the hypothesis is generated via the *variabilize*$(\cdot)$ operation. Once the hypothesis H is formed from S the algorithm asks an equivalence question: is H the correct expression? This is the main iteration structure which is repeated until the answer "yes" is obtained. On a positive counter-example (I is positive but $I \not\models H$), wrong clauses (s.t. $I \not\models C$) are removed from H as explained above.

On a negative counter-example (I is negative but $I \models H$), the algorithm first minimizes the number of objects in the counter-example using $I' = \textit{minimize-objects}(I)$. This is followed by generating a clause set $[s, c] = \textit{rel-cands}(I')$.

The algorithm then tries to find a "useful" pairing of $[s, c]$ with one of the clause sets $[s_i, c_i]$ in S. A useful pairing $[s_p, c_p]$ is such that s_p is a *negative* example also satisfying that s_p is *smaller* than s_i, where size is measured by the number of atoms in the set. The search is done by trying all possible matchings of objects in the corresponding clause sets and asking membership queries. The clause sets in S are tested in increasing order and the *first* $[s_i, c_i]$ for which this happens is replaced with the resulting pairing. The size constraint guarantees that measurable progress is made with each replacement. In case no such pairing is found for any of the $[s_i, c_i]$, the minimized clause set $[s, c]$ is added to S as the *last* element. Note that the order of elements in S is used in choosing the first $[s_i, c_i]$ to be replaced, and in adding the counter-example as the last element. These are crucial for the correctness of the algorithm.

Note that the algorithm enumerates matchings and corresponding pairings of clauses. As in the case of minimization the order of pairings considered can make a difference to the resulting hypothesis but the current system does not attempt to optimize this and uses some arbitrary order that is easy to calculate.

The interactive algorithm was previously analyzed and it has the following guarantees:

```
1. Initialize $S$ to be the empty sequence.
2. Repeat until $H \equiv T$:
   (a) Let $H = variabilize(S)$.
   (b) Ask an equivalence query to get a counter-example $I$ in case $H \not\equiv T$.
   (c) On a positive counter-example $I$ (s.t. $I \models T$):
         Remove wrong clauses (s.t. $I \not\models C$) from $H$.
   (d) On a negative counter-example $I$ (s.t. $I \not\models T$):
       i. $I' = minimize\text{-}objects(I)$.
       ii. $[s,c] = rel\text{-}cands(I')$.
       iii. For $i = 1$ to $m$ (where $S = ([s_1,c_1],\ldots,[s_m,c_m])$)
              For every pairing $[J,(c_i \cap c) \cup (s_i \setminus J)]$ of $[s_i,c_i]$ and $[s,c]$
              If $J$'s size is smaller than $s_i$'s size
                 and $J \not\models T$ (ask membership query) then
                A. Replace $[s_i,c_i]$ with $[J,(c_i \cap c) \cup (s_i \setminus J)]$.
                B. Quit loop (Go to Step 2a) .
       iv. If no $[s_i,c_i]$ was replaced then add $[s,c]$ as the last element of $S$.
```

Table 1: The Interactive Algorithm.

Theorem 1 (see Corollary 2 in Khardon (1999b)) *Assume there is a target expression T with m clauses each having at most k variables and that equivalence and membership queries are answered correctly according to T. Let p be the number of predicates in the signature, a the maximum arity and n the maximum number of objects in any negative counter-example provided to equivalence queries in the process of learning. Then the interactive algorithm will stop and produce a hypothesis equivalent to T with $O(mpk^ak^k)$ clauses after $O(mpk^ak^k)$ equivalence queries and $O((n+m^2)pk^ak^{3k})$ membership queries.*

3.3 The Batch Algorithm

The batch algorithm is based on the observation[2] that we can answer the interactive algorithm's questions using a given set of examples E.

Simulating equivalence queries is easy and is in fact well known. Given hypothesis H we evaluate H on all examples in E. If it misclassifies any example we have found a counter-example. Otherwise we found a consistent hypothesis. This procedure has statistical justification based in PAC learning theory (Angluin, 1988; Blumer et al., 1987). Essentially if we use a fresh sample for every query then with high probability a consistent hypothesis is good. If we use a single sample then by Occam's razor any short hypothesis is good.

For membership queries we use the following fact:

2. Similar observations were made by Kautz et al. (1995) and Dechter and Pearl (1992) with respect to the propositional algorithm of Angluin et al. (1992).

Lemma 2 (See Lemma 11 and 12 in Khardon 1999b) *Let T be a function free Horn expression and I an interpretation over the same alphabet. Then $I \not\models T$ if and only if for some $C \in$ rel-cands(I), $T \models C$.*

Now, we can simulate the test $T \models C$ by evaluating C on all positive examples in E. If we find a positive example e such that $e \not\models C$ then $T \not\models C$. Otherwise we assume that $T \models C$ and hence that $I \not\models T$.

A straightforward application will use the lemma directly whenever the algorithm asks a membership query (this is in fact algorithm *A4* of Khardon, 1999b). However, a more careful look reveals that queries will be repeated many times. Moreover, with large data sets, it is useful to reduce the number of passes over the data. We therefore optimize the procedure as described below.

The *one-pass* procedure: Given a clause set $[s,c]$ the procedure *one-pass* tests clauses in $[s,c]$ against all positive examples in E. The basic observation is that if a positive example can be matched to the antecedent but one of the consequents is false in the example under this matching then this consequent is wrong. For each positive example e, the procedure *one-pass* removes *all* wrong consequents identified by e from c. If c is empty at any point then we know that no atom is implied by s, that is, there are no correct consequents. Therefore the evaluation process is stopped and the procedure returns $[s,\emptyset]$ to indicate that this is the case.[3] At the end of *one-pass*, each consequent is correct w.r.t. the data set.

This operation is at the heart of the algorithm since the hypothesis and candidate clause sets are repeatedly evaluated against the data set. Two points are worth noting here. First, once we match the antecedent we can test all the consequents simultaneously so it is better to keep clause sets together rather than split them into individual clauses. Second, notice that since we must verify that consequents are correct, it is not enough to find just one substitution from an example to the antecedent. Rather we must check all such substitutions before declaring that some consequents are not contradicted. This issue affects the implementation and we discuss it further below.

Minimization: The minimization procedure acting on a clause set $[s,c]$ assumes the input clause set has already been validated by *one-pass*. It then iteratively tries to drop domain elements. In each iteration, it drops an object to get $[s',c']$ and runs *one-pass* on $[s',c']$ to get $[s'',c'']$. If c'' is not empty it continues with it to the next iteration (assigning $[s,c] \leftarrow [s'',c'']$); otherwise it continues with $[s,c]$. The final result of this process is $[s,c]$ in which all consequents are correct w.r.t. E.

The above simulation in *one-pass* avoids repeated queries that result from direct use of the lemma as well as guaranteeing that no positive counter-examples are ever found since they are used before clauses are put into the hypothesis. This simplifies the description of the algorithm which is summarized in Table 2.

3.4 Discussion of Batch Algorithm

The analysis of the interactive algorithm can be used to give some guarantees for the batch algorithm as well. In particular it is clear that all counter-examples used are correct since they are in the data set. If we can guarantee in addition that all membership queries implicit in *one-pass* are answered correctly then the batch algorithm can be seen as performing some run of the interactive algorithm and bounds on queries and hypothesis size translate as well.

3. Recall from above that the "empty consequent" is explicitly represented by `false` and if this is a correct consequent it will not be removed.

```
1. Initialize $S$ to be the empty sequence.
2. Repeat until $H$ is correct on all examples in $E$.
   (a) Let $H = variabilize(S)$.
   (b) If $H$ misclassifies $I$ ($I$ is negative but $I \models H$):
         i. $[s,c] = one\text{-}pass(rel\text{-}cands(I))$.
        ii. $[s,c] = minimize\text{-}objects([s,c])$.
       iii. For $i = 1$ to $m$ (where $S = ([s_1,c_1],\dots,[s_m,c_m])$)
              For every pairing $[J,(c_i \cap c) \cup (s_i \setminus J)]$ of $[s_i,c_i]$ and $[s,c]$
              If $J$'s size is smaller than $s_i$'s size then
                let $[s',c'] = one\text{-}pass([J,(c_i \cap c) \cup (s_i \setminus J)])$.
                If $c'$ is not empty then
                A. Replace $[s_i,c_i]$ with $[s',c']$.
                B. Quit loop (Go to Step 2a)
        iv. If no $[s_i,c_i]$ was replaced then add $[s,c]$ as the last element of $S$.
```

Table 2: The Batch Algorithm.

Recall that all of the membership queries asked by the interactive algorithm are sub-structures of negative examples. Consider a "complete" data set in the sense that every sub-structure of a negative example in the data set is also included in the data set. In this case the answer to a membership query on interpretation J exists explicitly in the data set. Similarly, the call to *one-pass* returns at least one conclusion (c' is not empty) if and only if a correct conclusion exists. In fact, a slightly weaker condition is also sufficient. We say that a data set is *relational complete for target T* if sub-structures of negative examples that are positive for T have a "representative" in the data where I' is a representative of I if it is isomorphic to I. Clearly such a data set provides exactly the same answer as a corresponding complete data set. We therefore have the following result:

Theorem 3 *Consider any data set which is relational complete for target T where T has m clauses each having at most k variables. Let p be the number of predicates in the signature, a the maximum arity and n the maximum number of objects in any negative example in the data set. Then the batch algorithm will stop and produce a hypothesis consistent with the data with $O(mpk^ak^k)$ clauses.*

Notice that the bound on hypothesis size does not depend on the number of examples but only on parameters of the target. If the data set is not complete the algorithm will still find a consistent hypothesis if one exists but the hypothesis may be large. While this notion of completeness is a strong condition to require, it can serve as a rough guide for data preparation and evaluating whether the algorithm is likely to work well for a given application. In some of the experiments below using artificial domains we generated the data in a way that is likely to include sub-structures of examples in other examples and indeed this led to good performance in these experiments.

The use of a data set to simulate queries raises a subtle aspect concerning time complexity. Assume the target expression T exists and that it uses at most k variables in each clause. It is shown in Khardon (1999b) that in such a case the minimization procedure outputs an interpretation with at most k domain elements. As a result any clause C produced by the interactive algorithm has at most k variables which in turn guarantees that we can test whether $I \models C$ in time $O(n^k)$ where I has n domain elements. However, for the batch algorithm we must simulate membership queries for the minimization process itself. When doing this we generate clauses with as many variables as there are domain elements in I, which may lead to the complexity of *one-pass* growing with n^n if examples typically have n domain elements. Moreover, as mentioned above we must enumerate all substitutions between a rule and positive examples in order to test whether we can remove conclusions. This is a serious consideration that might slow down the batch system in practice. It is therefore crucial for our system to have efficient procedures to test subsumption and enumerate matching. Several such methods are described below.

The same point also suggests that for some data sets the batch algorithm may overfit the data. In particular if we do not have sufficient positive examples to contradict wrong conclusions then we may drop the wrong objects in the process of minimization. As a result the rules generated may have a low correlation with the label. Since our algorithm depends on the order of examples, one way to reduce this effect is to sort the set of examples according to size. This is useful since smaller negative examples make less demands on the richness of the data set with respect to the *one-pass* procedure. If small examples contain seeds for all rules then this sidesteps the problem. In addition, sorting the examples by size helps reduce run time since the rules generated from the small examples have less variables, a property that generally implies faster subsumption tests. In the experiments described below we have sorted examples in this manner.

At this point it is interesting to compare the batch algorithm to the one used by the GOLEM system (Muggleton and Feng, 1992). The appendix discusses the relation between the so-called "normal ILP setting" used in GOLEM and the setting of learning from interpretations used by LOGAN-H. It suffices here to say that some manipulation allows us to work on the same problems. GOLEM uses bottom up learning by calculating the relative least general generalization (RLGG) of clauses. In our setting this corresponds to taking cross products of interpretations instead of pairings. This is incorporated in a greedy search that iteratively generalizes one of the clauses in its hypothesis by combining a new example with the clause. Thus the general structure of the algorithms is pretty similar. The main differences are that (1) we use pairings in order to get small clauses in the hypothesis whereas GOLEM uses the RLGG, that (2) pairing is aimed at learning multiple conclusions simultaneously thus moving atoms from antecedent to consequent, that (3) LOGAN-H uses the minimization procedure, and that (4) GOLEM greedily chooses the clause with maximum cover for combining clauses whereas LOGAN-H's choice is based on ordering the hypothesis clauses as suggested by the theoretical analysis.

3.5 Other Practical Considerations

The batch algorithm as described above includes all the main ideas but it may be hard to apply in some cases. The appendix gives details of heuristics implemented in the system that extend its applicability. In particular the system can handle noisy data (where no consistent hypothesis exists), handle numerical attributes through discretization, and use rule pruning. In addition the appendix describes several techniques that considerably improve run time. Aside from these, the most crucial run time issue is the subsumption test which we discuss next.

x_1	x_2	x_3	x_4
a	b		
a	c		
a	d		
b	a		
d	c		

x_1	x_2	x_3	x_4
a	b		
b	a		

x_1	x_2	x_3	x_4
a	b	b	
a	b	c	
a	b	d	
b	a	a	

x_1	x_2	x_3	x_4
a	b	b	a
a	b	d	c
b	a	a	b
b	a	a	c
b	a	a	d

Table 3: Example of Table Based Subsumption.

4. Efficient Subsumption Tests

The *one-pass* procedure must enumerate all substitutions that embed a clause in an example.[4] This problem is NP-Hard and hence we cannot expect a solution that is always efficient. Here we describe three different procedures that are shown to perform well in different settings; our system can work with any of these. This may be useful in other contexts as well.

4.1 Table Based Subsumption

While backtracking search (as done in Prolog) can find all substitutions without substantial space overhead, the time overhead can be very large. Our system implements an alternative approach that constructs all substitutions simultaneously and stores them in memory. The system maintains a table of instantiations for each predicate in the examples. To compute all substitutions between an example and a clause the system repeatedly performs joins of these tables (in the database sense) to get a table of all substitutions. We first initialize to an empty table of substitutions. Then for each predicate in the clause we pull the appropriate table from the example, and perform a join which matches the variables already instantiated in our intermediate table. Thus if the predicate in the clause does not introduce new variables the table size cannot grow. Otherwise the table can grow and repeated joins can lead to large tables. This approach is similar to the matching procedure of Di Mauro et al. (2003) and was developed independently; we discuss the similarities and differences in more detail below. To illustrate the method consider evaluating the clause $p(x_1,x_2), p(x_2,x_1), p(x_1,x_3), p(x_3,x_4)$ on an example with extension $[p(a,b), p(a,c), p(a,d), p(b,a), p(d,c)]$. Then applying the joins from left to right we get partial substitution tables given in Table 3 (from left to right). Notice how the first application simply copies the table from the extension of the predicate in the example. The first join reduces the size of the intermediate table. The next join expands both lines. The last join drops the row with `a b c` but expands other rows so that overall the table expands.

The code incorporates some heuristics to speed up computation. For example we check that each variable has at least one substitution common to all its instances in the clause. Otherwise we know that there is no substitution matching the clause to the examples. We also sort predicates in an example by their table size so that we first perform joins of small tables. We have also experimented with a module that performs lookahead to pick a join that produces the smallest table in the next step. This can substantially reduce the memory requirements and thus run time but on the other hand introduces overhead for the lookahead. Finally, taking inspiration from DJANGO (see description

4. It may be worth clarifying here that this is the standard subsumption problem and we do not require different variables to be matched to different objects. The one to one restriction appears in the analysis of Khardon (1999b) but is not part of the algorithm or its hypothesis.

below) we have also implemented a form of arc-consistency for the tables. In this case we first compute joins of table pairs and project them back onto the original variables. This removes rows that are clearly inconsistent from the tables. This is repeated until all tables are pairwise consistent and then the tables method is started as before.

One can easily construct examples where the table in intermediate steps is larger than the memory capacity of the computer, even if the final table is small. In this case the matching procedure will fail. This indeed occurs in practice and we have observed such large table sizes in the Mutagenesis domain (Srinivasan et al., 1994) as well as the artificial challenge problems of Giordana et al. (2003).

4.2 Randomized Table Based Subsumption

If lookahead is still not sufficient or too slow we can resort to randomized subsumption tests. Instead of finding all substitutions we try to sample from the set of legal substitutions. This is done in the following manner: if the size of the intermediate table grows beyond a threshold parameter TH (controlled by the user), then we throw away a random subset of the rows before continuing with the join operations. In this way we are not performing a completely random choice over possible substitutions. Instead we are informing the choice by our intermediate table. The absolute limit to the size of intermediate tables is TH$\times$16. If a join would produce a larger table we arbitrarily trim the table when it reaches this capacity, and then follow with the random selection. In addition the system uses random restarts to improve confidence as well as allowing more substitutions to be found. This can be controlled by the user through a parameter R.

4.3 Subsumption Based on Constraint Satisfaction Algorithms

The idea of using constraint satisfaction algorithms to solve subsumption problems has been investigated in Maloberti and Sebag (2004), where a very effective system DJANGO is developed. The DJANGO system was originally designed to find a single solution for the subsumption problem but this can be easily extended to give all solutions through backtracking. In the following we describe the ideas behind the original system that we refer to as DJANGODUAL as well as introduce a new method we call DJANGOPRIMAL that uses similar techniques but a different representation.
A Constraint Satisfaction Problem (CSP) (Tsang, 1993) is defined by a triplet $\langle \mathcal{V}, \mathcal{D}, \mathcal{C} \rangle$ such that:

$\mathcal{V}$ is a set of variables;

$\mathcal{D}$ is a set of domains, each domain D_V associates a set of admissible values to a variable $V \in \mathcal{V}$. The function $dom(V)$ returns the domain associated to the variable V.

$\mathcal{C}$ is a set of constraints, each constraint $C \in \mathcal{C}$ involves a tuple $\langle S, R \rangle$ such that :

- S is a subset of $\mathcal{V}$, denoted *scope* of a constraint. The function $vars(C)$ returns the variables in the scope of C, that is, $vars(C) = S$.
- R is a set of tuples, denoted *relations* of a constraint, each tuple is a set of simultaneously admissible values for each variable in S. The function $rel(C)$ returns the relations associated to the constraint C.

A constraint C is satisfied by an assignment of values to variables if and only if the corresponding tuple of assignments to $vars(C)$ is in $rel(C)$. A CSP is *satisfiable* iff it admits a solution, assigning to each variable $V \in \mathcal{V}$ a value $v \in dom(V)$ such that all constraints are satisfied. The arity of a CSP is the maximum number of variables in any constraint.

To illustrate how constraint satisfaction can be used consider the following subsumption problem where the clause is Cl and the example is Ex:

$$\begin{array}{rl} Cl: & p(X_0,X_1), q(X_0,X_2,X_3), r(X_0), \\ Ex: & p(a_0,a_1), p(a_1,a_2), q(a_0,a_2,a_3), q(a_0,a_1,a_3), r(a_0). \end{array}$$

This problem can be trivially transformed into a CSP problem such that:

- The set of variables $\mathcal{V}$ is equivalent to the set of variables of Cl; $\{X_0, X_1, X_2, X_3\}$;
- All domains are the same and are equivalent to the set of objects in Ex, $D = \{a_0, a_1, a_2, a_3\}$;
- Each literal l_i in Cl is transformed into a constraint C_i such that:
  - the scope S_i of C_i corresponds to the set of variables of l_i;
  - the relation R_i of C_i includes all variable tuples from instances of l_i's predicate in Ex.

  Therefore, there are three constraints C_1, C_2 and C_3 in our example, which respectively account for the literals $r(X_0)$, $p(X_0,X_1)$ and $q(X_0,X_2,X_3)$, such that:

  $S_1 = \{X_0\}$ and $R_1 = \{\langle a_0 \rangle\}$,
  $S_2 = \{X_0,X_1\}$ and $R_2 = \{\langle a_0, a1\rangle, \langle a_1,a_2\rangle\}$,
  $S_3 = \{X_0,X_2,X_3\}$ and $R_3 = \{\langle a_0,a_2,a_3\rangle, \langle a_0,a_1,a_3\rangle\}$.

Thus, finding an assignment for all variables of this CSP which satisfy all the constraints, is equivalent to finding a substitution θ such that $\mathcal{C}\theta \subseteq Ex$.

In order to distinguish this representation from others, this CSP is called the *Primal representation*.

Since an n-ary CSP can always be transformed into a binary CSP, that is, a CSP such that all constraints involve at most 2 variables, most algorithms in CSP are restricted to binary CSPs. Thus, in order to simplify the implementation of DJANGODUAL, a transformation named *dualization* was used. To avoid confusion we call variables in the dual representation Δ-variables. Dualization encapsulates each constraint of the primal CSP into a Δ-variable, and a constraint is added in the dual CSP between each pair of Δ-variables which shares at least one variable.

Therefore, the problem of the previous example is transformed into a dual CSP, where:

- Each literal l in $\mathcal{C}$ gives rise to a *dual variable* Δ_l. Therefore, $\mathcal{V} = \{\Delta_r, \Delta_p, \Delta_q\}$. If more than one literal has the same predicate symbol p, Δ-variables are denoted $\{\Delta_{p.1}, \Delta_{p.2}, \ldots, \Delta_{p.n}\}$.
- Each domain of a Δ-variable Δ_l is equivalent to the set of literals in Ex with the same predicate symbol as l, $dom(\Delta_r) = \{r(a_0)\}$, $dom(\Delta_p) = \{p(a_0,a_1), p(a_1,a_2)\}$ and $dom(\Delta_q) = \{q(a_0,a_2,a_3), q(a_0, a_1, a_3)\}$.
- A constraint is created for each pair of literals which share at least one variable in $\mathcal{C}$. The relation R_i contains all pairs of literals in Ex in which the corresponding primal variables are given the same values (so they are compatible). Since X_0 is shared by all the literals, there are three constraints C_1, C_2 and C_3 in our example, where:

$$S_1 = \{\Delta_r, \Delta_p\}, R_1 = \{\langle r(a_0), p(a_0,a_1)\rangle\},$$
$$S_2 = \{\Delta_r, \Delta_q\}, R_2 = \{\langle r(a_0), q(a_0,a_2,a_3)\rangle, \langle r(a_0), q(a_0,a_1,\, a_3)\rangle\},$$
$$S_3 = \{\Delta_p, \Delta_q\}, R_3 = \{\langle p(a_0,a_1), q(a_0,a_2,a_3)\rangle, \langle p(a_0,a_1), q(a_0,a_1,a_3)\rangle\}.$$

In the particular case of multiple occurrences of a variable in the same literal, a unary constraint is created in order to check that all literals in the domain also have the same term in corresponding positions.

Dualization performs $\frac{n\times(n-1)}{2}$ comparisons of literals, where n is the number of literals in $\mathcal{C}$. Each comparison of literals involves a^2 tests of equivalence between variables, where a is the maximal arity of the predicates. Thus, building a dual CSP from a clause is obviously a polynomial transformation. It does however incur an overhead before starting to solve the subsumption problem and can represent an important part of the execution time in case of easy instances of subsumption problems.

It has been shown (Chen, 2000) that polynomially solvable instances of Primal CSPs may require exponential time in the Dual representation, and vice versa. The system of Maloberti and Sebag (2004) uses the dual representation and we therefore refer to it as DJANGODUAL. LOGAN-H includes both options and in particular we have implemented a new solver using the primal representation, which we call DJANGOPRIMAL. Despite the different representations, both versions use a similar method to solve the subsumption problem, using arc-consistency and following with depth first search with dynamic variable ordering. However, due to the fact that the primal representation has non binary constraints, simpler heuristics are used in DJANGOPRIMAL. Some implementation details and the differences between the two methods are described in the appendix.

4.4 Discussion of Subsumption Methods

The main difference between the table based method and the DJANGO methods is that all substitutions are calculated simultaneously in the table method and by backtracking search in DJANGO. This has major implications for run time. In cases where the tables are applicable and where the intermediate tables are small the tables can save a substantial amount of overhead. On the other hand the tables can get too large to handle. Even when tables are of moderate size, memory copying operations when handling joins can be costly so we are introducing another overhead. Within LOGAN-H the backtracking approach can benefit when we have many substitutions of which only a few are needed to remove all consequents from a clause set.

Di Mauro et al. (2003) have previously introduced a table based method for subsumption tests (although our system was developed independently). The main difference between the two approaches is that their method tries to optimize space by compressing table entries. Essentially, their tables try to compress multiple substitutions into a single line. This can be done whenever variables have independent constraints. Therefore, one may expect their method to do better in terms of space when tables are large, but possibly at the cost of run time to compress and decompress the substitutions. In contrast, our randomized method attempts to save both time and space but this is at the cost of completeness.

The table-based subsumption method incorporates aspects from both the primal and dual representation of DJANGO. The way partial substitutions are represented as tables is similar to the dual variables and the method can be seen as reducing the number of dual variables through the join operations. The test that guarantees that each variable has at least one potential value to match is related to arc-consistency in the primal representation since it works by reducing domains of primal variables. On the other hand the arc-consistency improvement is done over the tables, that is in the dual representation.

Finally, calculating the dual form representation of the subsumption problem incurs some overhead that can be noticeable if the subsumption problems themselves are quickly solved. Since in LOGAN-H we have many subsumption tests with the same example or the same clause these can be avoided with additional caching. This gives a substantial improvement for the overall run time of LOGAN-H when using DJANGODUAL.

Other systems have developed different methods to reduce the computational cost of subsumption tests. Blockeel et al. (2000) and Blockeel et al. (2002) propose the use of *query packs*. Query packs make the evaluation of a set of hypotheses against a clause more efficient. A query-pack structures a set of hypotheses in a tree where each node is a set of literals, hypotheses with common parts have a common path to the root. Therefore, common parts are evaluated only once for all the hypotheses instead of once for each hypothesis. The query-packs can drastically improve performance, particularly with top-down algorithms, since they usually create new hypotheses by adding new literals to a candidate hypothesis, which is therefore common to all new hypotheses. Santos Costa et al. (2003) propose the combined use of several transformations based on additional information on the variable types to speed up subsumption. This additional information allows to recursively decompose the clause into independent parts up to the instantiation of "grounded variables". These techniques are orthogonal to ours so that it may be possible to incorporate them for further performance improvement.

5. Experimental Evaluation

The ideas described above have been implemented in two different systems. The first one, initially reported in Khardon (2000), implements the interactive and batch algorithms in the Prolog language but does not include discretization, pruning or special subsumption engines. The second implementation, recently reported in Arias and Khardon (2004), was done using the C language and implements only the batch algorithm but otherwise includes all the techniques described above.

In this section we describe several experiments using the system and its subsumption engines. The experiments are designed to test and demonstrate several issues. First, they exemplify the scope and type of applications that may be appropriate. Second, they demonstrate that LOGAN-H can give state of the art performance on relational learning problems, and that it results in very strong hypotheses when the data has some similarity to being "complete" in the sense discussed above. Third, the experiments evaluate the contribution of different speedup techniques in the system. Fourth, the experiments evaluate the performance of different subsumption engines both within the learning system and independently.

The experiments are as follows. We first illustrate the scope and limitation of interactive algorithm on the task of learning complex list manipulation programs. We then discuss a toy grammar learning problem showing that the batch algorithm can be applied to this potentially hard problem using carefully selected data. We then describe a set of quantitative experiments in three benchmark ILP domains: the Bongard domain (De Raedt and Van Laer, 1995), the KRK-illegal domain (Quinlan, 1990), and the Mutagenesis domain (Srinivasan et al., 1994). The experiments illustrate applicability in terms of scalability and accuracy of the learned hypotheses as well as providing a comparison with other systems. They also demonstrate that different subsumption engines may lead to faster execution in different problems.

We also compare the subsumption methods in an experiment based on the Phase Transition phenomenon similar to the experiments performed by Maloberti and Sebag (2004). These experiments show that DJANGOPRIMAL is very effective and may perform better than the other two approaches if hard subsumption problems around the phase transition region need to be solved.

5.1 Experiments with LOGAN-H

This section describes our experiments with LOGAN-H in the 5 domains as outlined above.

5.1.1 LEARNING LIST MANIPULATION PROGRAMS

To facilitate experiments using the interactive mode, we have implemented an "automatic-user mode" where (another part of) the system is told the expression to be learned (the target T). The algorithm's questions are answered automatically using T. Since implication for function-free Horn expressions is decidable this can be done reliably. In particular, Lemma 13 in Khardon (1999b) provides an algorithm that can test implication and construct counter-examples to equivalence queries. Membership queries can be evaluated on T. We note that this setup is generous to our system since counter-examples produced by the implemented "automatic user mode" are in some sense the smallest possible counter-examples.

Using this mode we ran the interactive algorithm to learn a 15-clause program including a collection of standard list manipulation procedures. The program includes: 2 clauses defining $list(L)$, 2 clauses defining $member(I,L)$, 2 clauses defining $append(L1,L2,L3)$, 2 clauses defining $reverse(L1,L2)$, 3 clauses defining $delete(L1,I,L2)$, 3 clauses defining $replace(L1,I1,I2,L2)$, and 1 clause defining $insert(L1,I,L2)$ (via $delete()$ and $cons()$). The definitions have been appropriately flattened so as to use function free expressions. All these clauses for all predicates are learned simultaneously. The Prolog system required 35 equivalence queries, 455 membership queries and about 8 minutes (running Sicstus Prolog on a Linux platform using a Pentium 2/366MHz processor) to recover the set of clauses exactly.[5] This result is interesting since it shows that one can learn a complex program defining multiple predicates in interactive mode with a moderate number of questions. However, the number of questions is probably too large for a program development setting where a human will answer the questions. It would be interesting to explore the scope for such use in a real setting.

5.1.2 A TOY GRAMMAR LEARNING PROBLEM

Consider the problem of learning a grammar for English from examples of parsed sentences. In principle, this can be done by learning a Prolog parsing program. In order to test this idea we generated examples for the following grammar, which is a small modification of one described by Pereira and Shieber (1987).

```
s(H2,P0,P)  :- np(H1,P0,P1),vp(H2,P1,P),number(H1,X),number(H2,X).
np(H,P0,P)  :- det(HD,P0,P1),n(H,P1, P),number(HD,X),number(H,X).
np(H,P0,P)  :- det(HD,P0,P1),n(H,P1, P2),rel(H2,P2,P),number(H,X),
```

5. This formalization is not range-restricted so we did not use the restriction to live pairings given in the appendix.

```
                number(H2,X),number(HD,X).
np(H,P0,P)  :- pn(H,P0,P).
vp(H,P0,P)  :- tv(H,P0,P1),np(_,P1,P).
vp(H,P0,P)  :- iv(H,P0,P).
rel(H,P0,P) :- relw(_,P0,P1),vp(H,P1,P,L1).
```

Note that the program corresponds to a chart parser where we identify a "location parameter" at beginning and end of a sentence as well as between every two words, and true atoms correspond to edges in the chart parse. Atoms also carry "head" information for each phrase and this is used to decide on number agreement. The grammar allows for recursive sub-phrases. A positive example for this program is a sentence together with its chart parse, that is, all atoms that are true for this sentence. The base relations identifying properties of words, $det(), n(), pn(), iv(), tv(), relw(), number()$, are assumed to be readable from a database or simply listed in the examples.

We note that the grammar learning problem as formalized here is interesting only if external properties such as *number*() are used. Otherwise, one can read-off the grammar rules from the structure of the parse tree. It is precisely because such information is important for the grammar but normally not supplied in parsed corpora that this setup may be useful. Of course, it is not always known which properties are the ones crucial for correctness of the rules as this implies that the grammar is fully specified. In order to model this aspect in our toy problem we included all the relations above and in addition a spurious property of words *confprop*() (for "confuse property") whose values were selected arbitrarily. For example a chart parse of the sentence "sara writes a program that runs" is represented using the positive example:

```
([sara,writes,a,program,that,runs,alpha,beta,singular,0,1,2,3,4,5,6],
 [s(writes,0,4), s(writes,0,6),
  np(sara,0,1), np(program,2,4), np(program,2,6),
  vp(writes,1,4), vp(runs,5,6), vp(writes,1,6),
  rel(runs,4,6), pn(sara,0,1), n(program,3,4), iv(runs,5,6),
  tv(writes,1,2), det(a,2,3), relw(that,4,5),
  number(runs,singular), number(program,singular), number(a,singular),
  number(writes,singular), number(sara,singular),
  confprop(runs,beta), confprop(program,alpha),
  confprop(writes,beta),confprop(sara,alpha)
 ]).
```

Note that one can generate negative examples from the above by removing implied atoms of $s(), np(), vp(), rel()$ from the interpretation. It may be worth emphasizing here that negative examples are not non-grammatical sentences but rather partially parsed strings. Similarly, a non-grammatical sequence of words can contribute a positive example if all parse information for it is included. For example "joe joe" can contribute the positive example

```
([joe,0,1,2,alpha,singular],
 [pn(joe,0,1), pn(joe,1,2), np(joe,0,1), np(joe,1,2),
  number(joe,singular),confprop(joe,alpha)
 ]).
```

or a negative one such as

```
([joe,0,1,2,alpha,singular],
 [pn(joe,0,1), pn(joe,1,2), np(joe,0,1),
  number(joe,singular),confprop(joe,alpha)
 ]).
```

A simple analysis of the learning algorithm and problem setup shows that we must use non-grammatical sentences as well as grammatical ones and we have done this in the experiments. For example, if all examples have agreement in number, the algorithm has no way of finding out whether the *number*() atoms in an example can be dropped or not since nothing would contradict dropping them.

A final issue to consider is that for the batch algorithm to work correctly we need to include positive sub-structures in the data set. While it is not possible to take all sub-structures we approximated this by taking all *continuous* substrings. Given a sentence with k words, we generated from it all $O(k^2)$ substrings, and from each we generated positive and negative examples as described above.

We ran two experiments with this setup. In the first we hand picked 11 grammatical sentences and 8 non-grammatical ones that "exercise" all rules in the grammar. With the arrangement above this produced 133 positive examples and 351 negative examples. The batch algorithm recovered an exact copy of the grammar, making 12 equivalence queries and 88 calls to *one-pass*.

In the second experiment we produced all grammatical sentences with "limited depth" restricting arguments of *tv*() to be *pn*() and allowing only *iv*() in relative clauses. This was done simply in order to restrict the number of sentences, resulting in 120 sentences and 386 sub-sentences. We generated 614 additional random strings to get 1000 base strings. These together generated 1000 positive examples and 2397 negative examples. With this setup the batch algorithm found a hypothesis consistent with all the examples, using 12 equivalence queries and 81 calls to *one-pass*. The hypothesis included all correct grammar rules plus 2 wrong rules. This is interesting as it shows that, although some examples are covered by wrong rules, other examples reintroduced seeds for the correct rules and then succeeded in recovering the rules.

The experiments demonstrate that it is possible to apply our algorithms to problems of this type even though the setup and source of examples is not clear from the outset. They also show that it may be possible to apply our system (or other ILP systems) to some NLP problems but that data preparation will be an important issue in such an application.

5.1.3 BONGARD PROBLEMS

The Bongard domain is an artificial domain that was introduced with the ICL system (De Raedt and Van Laer, 1995) to test systems that learn from interpretations. In this domain an example is a "picture" composed of objects of various shapes (triangle, circle or square), triangles have a configuration (up or down) and each object has a color (black or white). Each picture has several objects (the number is not fixed) and some objects are inside other objects. For our experiments we generated random examples, where each parameter in each example was chosen uniformly at random. In particular we used between 2 and 6 objects, the shape color and configuration were chosen uniformly at random, and each object is inside some other object with probability 0.5 where the target was chosen uniformly at random among "previous" objects to avoid cycles. Note that since we use a function free representation the domain size in examples is larger than the number of objects (to include: *up*, *down*, *black*, *white*). As in the previous experiment, this mode of data generation has

Target	*Clauses*	*Atoms*	*Variables*
I	2	4	2
II	2	6	4
III	2	9	6
IV	10	9	6

Table 4: Complexity of Targets.

System	Target	200	500	1000	2000	3000	Majority Class
LOGAN-H	I	99.7	100	100	100	100	64.4
LOGAN-H	II	97.6	99.4	99.9	99.9	100	77.8
LOGAN-H	III	90.8	97.2	99.3	99.9	99.9	90.2
LOGAN-H	IV	85.9	92.8	96.8	98.4	98.9	84.7
ICL	IV	85.2	88.6	89.1	90.2	90.9	84.7

Table 5: Performance summary in average percentage accuracy.

some similarity to the "closure" property of data sets that guarantees good performance with our algorithm.

In order to label examples we arbitrarily picked 4 target Horn expressions of various complexities. Table 4 gives an indication of target complexities. The first 3 targets have 2 clauses each but vary in the number of atoms in the antecedent and the fourth one has 10 clauses of the larger kind making the problem more challenging. The numbers of atoms and variables are meant as a rough indication as they vary slightly between clauses. To illustrate the complexity of the targets, one of the clauses in target IV is

$$\begin{array}{l} circle(X)\ in(X,Y)\ in(Y,Z)\ color(Y,B) \\ color(Z,W)\ black(B)\ white(W)\ in(Z,U) \quad \rightarrow \quad triangle(Y) \end{array}$$

We ran the batch algorithm on several sample sizes. Table 5 summarizes the accuracy of learned expressions as a function of the size of the training set (200 to 3000) when tested on classifying an independent set of 3000 examples. Each entry is an average of 10 independent runs where a fresh set of random examples is used in each run. The last column in the table gives the majority class percentage.

Clearly, the algorithm is performing very well on this problem setup. Note that the baseline is quite high, but even in terms of relative error the performance is good. We also see a reasonable learning curve obtained for the more challenging problems. Notice that for target I the task is not too hard since it is not unlikely that we get a random example matching the antecedent of a rule exactly (so that discovering the clause is easy) but for the larger targets this is not the case. We have also run experiments with up to 10 shapes per example with similar performance.

To put these experiments in perspective we applied the systems ICL (De Raedt and Van Laer, 1995) and Tilde (Blockeel and De Raedt, 1998) to the same data. Exploratory experiments suggested that ICL performs better on these targets so we focus on ICL. We ran ICL to learn a Horn CNF and otherwise with default parameters. ICL uses a scheme of declarative bias to restrict its search space.

With a general pattern implying little bias, success was limited. We thus used a bias allowing up to 4 shapes and identifying the relation between *config* and *up*, *down* and similarly *color* and *black*, *white*. Interestingly, for ICL the change in performance from target I to IV was less drastic than in LOGAN-H. This may well be due to the fact that LOGAN-H builds antecedents directly from examples. The last line in Table 5 gives the performance of ICL on target IV. As can be seen our system performs better than ICL on this problem.

We ran the Prolog implementation (using compiled code in Sicstus Prolog) and the new C implementation on the same hardware and observed a speedup of over 400-fold when using the tables method or DJANGODUAL and a speedup of 320 when using DJANGOPRIMAL. Recall that the number of objects in the examples is relatively small for this experiment. For larger examples as in the experiments that follow the improvement is even more dramatic as the Prolog code cannot complete a run and the C code is still pretty fast.

5.1.4 ILLEGAL POSITIONS IN CHESS

Our next experiment is in the domain of the chess endgame White King and Rook vs. Black King. The task is to predict whether a given board configuration represented by the 6 coordinates of the three chess pieces is illegal or not. This learning problem has been studied by several authors (Muggleton et al., 1989; Quinlan, 1990). The data set includes a training set of 10000 examples and a test set of the same size.

We use the predicate `position(a,b,c,d,e,f)` to denote that the White King is in position $(\mathtt{a},\mathtt{b})$ on the chess board, the White Rook is in position $(\mathtt{c},\mathtt{d})$, and the Black King in position $(\mathtt{e},\mathtt{f})$. Additionally, the predicates "less-than" `lt(x,y)` and "adjacent" `adj(x,y)` denote the relative positions of rows and columns on the board. Note that there is an interesting question as how best to capture examples in interpretations. In "all background mode" we include all `lt` and `adj` predicates in the interpretation. In the "relevant background mode" we only include those atoms directly relating objects appearing in the position atom.

We illustrate the difference with the following example. Consider the configuration "White King is in position (7,6), White Rook is in position (5,0), Black King is in position (4,1)" which is illegal. In "all background mode" we use the following interpretation:

```
[position(7, 6, 5, 0, 4, 1),
lt(0,1), lt(0,2), ..  ,lt(0,7),
lt(1,2), lt(1,3), ..  ,lt(1,7),
⋮
lt(5,6),lt(5,7),
lt(6,7),
adj(0,1),adj(1,2), ..  ,adj(6,7),
adj(7,6),adj(6,5), ..  ,adj(1,0)]-
```

When considering the "relevant background mode", we include in the examples instantiations of `lt` and `adj` whose arguments appear in the position atom directly:

```
[position(7, 6, 5, 0, 4, 1),
lt(4,5),lt(4,7),lt(5,7),adj(4,5),adj(5,4),
lt(0,1),lt(0,6),lt(1,6),adj(0,1),adj(1,0)]-
```

Table 6 includes results of running our system in both modes. We trained LOGAN-H on samples with various sizes chosen randomly among the 10000 available. We report accuracies that result

	25	50	75	100	200	500	1000	2000	3000
Relevant background mode:									
LOGAN-H before pruning	75.49	88.43	93.01	94.08	97.18	99.54	99.79	99.92	99.96
LOGAN-H after pruning	86.52	90.92	94.19	95.52	98.41	99.65	99.79	99.87	99.96
All background mode:									
LOGAN-H before pruning	67.18	71.08	75.71	78.94	85.56	94.06	98.10	99.38	99.56
LOGAN-H after pruning	79.01	81.65	83.17	82.82	86.02	93.67	96.24	98.10	98.66
FOIL (Quinlan, 1990)				92.50			99.40		

Table 6: Performance summary for KRK-illegal data set in average percentage accuracy

Subsumption Engine	*runtime in s.*	*accuracy*	*actual table size*
DJANGODUAL	40.27	98.10%	n/a
DJANGOPRIMAL	78.74	98.10%	n/a
Tables	136.43	98.10%	130928
Tables ARC Consistency	92.47	98.10%	109760
Lookahead	191.50	98.10%	33530
No cache	503.92	98.10%	130928
Rand. TH=1	3804.52	33.61%	16
Rand. TH=10	178.69	33.61%	160
Rand. TH=100	58.41	72.04%	1600
Rand. TH=1000	126.48	98.10%	16000

Table 7: Runtime comparison for subsumption tests on KRK-illegal data set

from averaging among 10 runs over an independent test set of 10000 examples. Results are reported before and after pruning where pruning is done using the training set (as discussed in the appendix). Several facts can be observed in the table. First, we get good learning curves with accuracies improving with training set size. Second, the results obtained are competitive with results reported for FOIL (Quinlan, 1990). Third, relevant background knowledge seems to make the task easier. Fourth, pruning considerably improves performance on this data set especially for small training sets.

This domain is also a good case to illustrate the various subsumption tests in our system. Note that since we put the position predicate in the antecedent, the consequent is nullary so iterative subsumption tests are likely to be faster. The comparison is given for the "all background mode" with 1000 training examples. Table 7 gives accuracy and run time (on Linux running with Pentium IV 2.80 GHz) for various subsumption settings averaged over 10 independent runs. For randomized runs TH is the threshold of table size after which sampling is used. As can be seen DJANGODUAL is faster than DJANGOPRIMAL in this domain and both are faster than the tables method. Adding arc-consistency to the tables method improves both space requirements and run time. The lookahead table method incurs some overhead and results in slower execution on this domain, however it saves space considerably (see third column of Table 7). The caching mechanism described in the appendix gives a significant reduction in run time. Running the randomized test with very small tables (TH=1) clearly leads to overfitting, and in this case increases run time considerably mainly

Method	Avg. accuracy
Equal Frequency best split `atom-charge`=25 `lumo`=8 `logp`=30	89.41%
Equal Frequency automatic selection	83.62%
Information Gain	82.25%

Table 8: Accuracy results on the Mutagenesis domain.

due do the large number of rules induced. On the other hand with larger table sizes (TH=1000) the randomized method does very well and reproduces the deterministic results.

5.1.5 MUTAGENESIS

The Mutagenesis data set is a structure-activity prediction task for molecules introduced by Srinivasan et al. (1994). The data set consists of 188 compounds, labeled as active or inactive depending on their level of mutagenic activity. The task is to predict whether a given molecule is active or not based on the first-order description of the molecule. This data set has been partitioned into 10 subsets for 10-fold cross validation estimates and has been used in this form in many studies (Srinivasan et al., 1994; Sebag and Rouveirol, 2000; De Raedt and Van Laer, 1995). For the sake of comparison we use the same partitions as well. Each example is represented as a set of first-order atoms that reflect the atom-bond relations of the compounds as well as some interesting global numerical chemical properties and some elementary chemical concepts such as aromatic rings, nitro groups, etc. Concretely, we use all the information corresponding to the background level B3 of Srinivasan et al. (1995). Notice that the original data is given in the normal ILP setting and hence we transformed it according to the interpretations setting as explained in the appendix. In addition, since constants are meaningful in this data set, for example whether an atom is a carbon or oxygen, we use a flattened version of the data where we add a predicate for each such constant.

This example representation uses continuous attributes: `atom-charge`, `lumo` and `logp` and hence discretization is needed. We report on experiments with the equal frequency method (Dougherty et al., 1995) and the information gain (IG) method (Fayyad and Irani, 1993) which are discussed in more detail in the appendix. The IG method decides on the number of bins automatically. The equal frequency method requires the number of bins as input. To get reliable results we used double cross validation performing automatic parameter selection over the following range of value: for `atom-charge` $\{5,15,25,35,45\}$ for `lumo` $\{4,6,8,10,20,30,80\}$ and for `logp` $\{4,6,8,10,30,50\}$. For reference we also report the best result that can be obtained in hindsight by searching for good partitions using the test set.

Table 8 summarizes some of the accuracies obtained. As can be seen the overoptimistic "best" results give pretty high accuracy. The automatic selection methods perform quite well giving 83.62% and the IG method comes close with 82.25%.

Our result compares well to other ILP systems: Progol (Srinivasan et al., 1994) reports a total accuracy of 83% with B3 and 88% with B4; STILL (Sebag and Rouveirol, 2000) reports results in the range 85%–88% on B3 depending on the values of various tuning parameters, ICL (De Raedt and Van Laer, 1995) reports an accuracy of 84% and finally Laer et al. (1996) report that FOIL (Quinlan, 1990) achieves an accuracy of 83%.

Subsumption Engine	*runtime*	*accuracy*
DJANGODUAL	3.48 *sec.*	89.41%
DJANGOPRIMAL	0.82 *sec.*	89.41%
Rand. TH=1 R=1	0.74 *sec.*	88.88%
Rand. TH=10 R=1	0.93 *sec.*	90.46%
Rand. TH=100 R=1	3.01 *sec.*	90.46%
Rand. TH=1 R=10	0.91 *sec.*	90.46%
Rand. TH=1 R=100	2.92 *sec.*	89.93%
Rand. TH=10 R=100	9219.68 *sec.*	89.93%

Table 9: Runtime comparison for subsumption tests on Mutagenesis data set.

We have also run experiments comparing run time with the different subsumption engines. For this domain, deterministic table-based subsumption was not possible, not even with lookahead and arc-consistency since the table size grew beyond memory capacity of our computer. However, both modes of DJANGO are very efficient on this domain. For these runs we used the equal frequency discretization method with `atom-charge`= 25, `lumo`= 8 and `logp`= 30 that gave top-scoring accuracy as described above. Table 9 gives average run time (on Linux running with a 2.80 GHz Xeon processor) per fold as well as the average accuracy obtained. One can observe that DJANGOPRIMAL is faster than DJANGODUAL and that even with small parameters the randomized methods do very well. An inspection of the hypothesis to the deterministic runs shows that they are very similar.

5.2 Subsumption and Phase Transition

The previous experiments have demonstrated that different subsumption engines may lead to faster performance in different domains. In this section we further compare the different algorithms but purely on subsumption problems, that is, not in the context of learning. Previous work (Giordana and Saitta, 2000) has shown that one can parameterize subsumption problems so that there is a sharp transition between regions where most problems have a solution and regions where most problems do not have a solution. This is known as the phase transition phenomenon, and it has been used to evaluate subsumption and learning algorithms (Giordana and Saitta, 2000; Giordana et al., 2003; Maloberti and Sebag, 2004).

For these experiments, a set of clauses and a set of examples are generated using four parameters:

n The number of variables in each clause, which is set to 12;

m The number of literals in each clause, which varies in $[12,50]$;

N The number of literals built on each predicate symbol in each example, that is, the size of each domain in dual representation, which is set to 50;

L The number of constants in each example, that is, the size of each domain in primal representation, which varies in $[12,50]$.

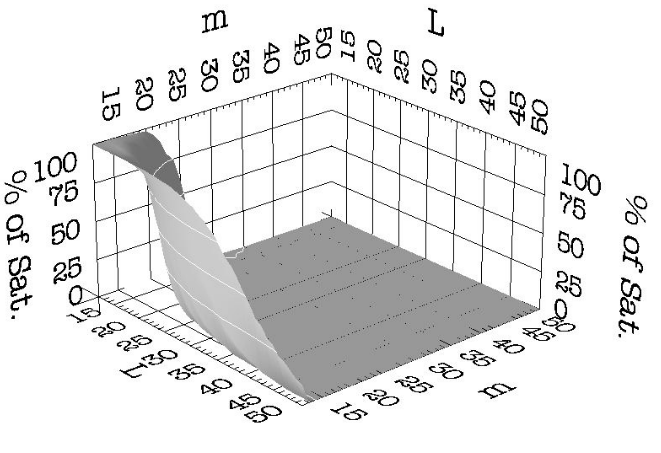


Figure 2: Percentage of subsumption tests satisfied on 50×50 pairs $(\mathcal{C}, Ex)$, where $\mathcal{C}$ is a clause uniformly generated with $n = 12$ variables and m literals (m in [12,50]), and Ex is an example uniformly generated with $N = 50$ literals built on each one of the m predicate symbols in $\mathcal{C}$, and L constants (L in [12,50]).

All m literals in a generated clause $\mathcal{C}$ are built on distinct binary predicate symbols and clause $\mathcal{C}$ is connected, that is, all n variables are linked. The latter requirement prevents the subsumption problem from being decomposable into simpler problems.

Each generated example Ex is a conjunction of ground literals. Each literal in Ex is built on a predicate symbol occurring in $\mathcal{C}$ (other literals are irrelevant to the subsumption problem). The number of literals in Ex per predicate symbol, N, is constant, thus all domains of the literals of $\mathcal{C}$ have the same size, and each example contains $N \times m$ literals. For each pair of values of $\langle m, L\rangle$, 50 clauses and 50 examples are generated, each clause is tested against all examples. Run times are measured on an Athlon 900 MHz with 512MB of memory.[6]

As shown in Figure 2, the probability for $\mathcal{C}$ to subsume Ex abruptly drops from 100% to 0% in a very narrow region. This region is called Phase Transition, and is particularly important for the subsumption problem, since computationally hard problems are located in this region. This phenomenon can be observed in Figure 3 (A) representing the average computational costs for DJANGODUAL. Figure 3 (B) shows the average execution times for DJANGOPRIMAL. The Phase

6. The experiments performed here are slightly different from the ones in Maloberti and Sebag (2004). First, the computational cost of the transformation to the dual representation is also measured here. This is important in the context of a system that dynamically builds clauses and tests subsumption for them. Although the cost of dualization is generally moderate, it can be significant in some experiments so it must be included in the execution time. Second, settings used in former experiments were: $N = 100, n = 10, L$ and m in $[10, 50]$. In the dual representation, the worst case complexity is N^m, while in primal representation it is L^n. Therefore, $N = 100$ is too large compared to L, and $n = 10$ is too small compared to m. It is difficult to change these settings because a slight variation can shift the phase transition to be outside of reasonable ranges of values. Thus, we reduced N to 50 and adjusted n to 12. L and m have been set to $[12, 50]$ in order to keep the property of non-decomposability of $\mathcal{C}$. Finally, in Maloberti and Sebag (2004), 100 clauses and 100 examples were generated, each clause was tested against one example.

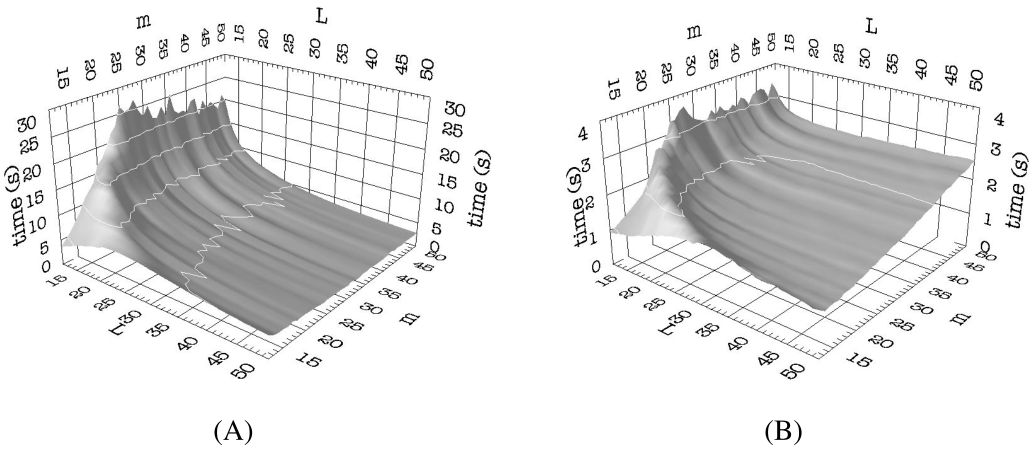


Figure 3: Subsumption cost(m,L) in seconds for (A) DJANGODUAL and (B) DJANGOPRIMAL averaged over 50×50 pairs $(\mathcal{C}, Ex)$. A reduction in running time of a factor of 7 is observed in DJANGOPRIMAL compared to Meta-DJANGO (notice difference in scale of the time axis).

Transition phenomenon is also apparent, however DJANGOPRIMAL is 7 times faster than DJANGODUAL on average, despite the simpler versions of algorithms used in DJANGOPRIMAL.

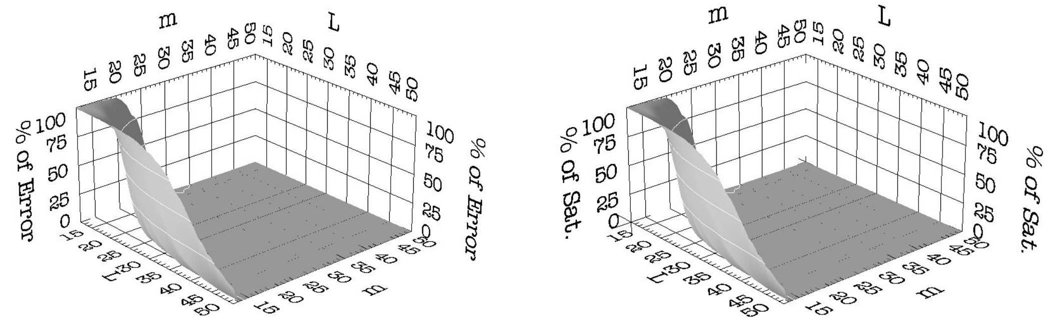
The table based methods are less well suited for the problems in this test suite. The deterministic table methods ran out of memory and could not be used. The randomized methods give a tradeoff between run time and accuracy but they perform worse than DJANGO.

For example, the randomized tables method with TH= 1 and R= 1 is very fast with an average execution time between $0.37s$ and $0.54s$. However, almost no solution can be found, even in the region of the test with a very high satisfiability. Therefore, as shown in Figure 4 (A), its error rate is close the percentage of satisfiability.

As shown in Figure 5 (A), the randomized tables method with TH= 10 and R= 10 is slower than DJANGO, and its average cost does not rely on the phase transition. On the other hand, Figure 5 (B) shows that its error rate is very high in the phase transition region, while it is now very low in the region with high satisfiability.

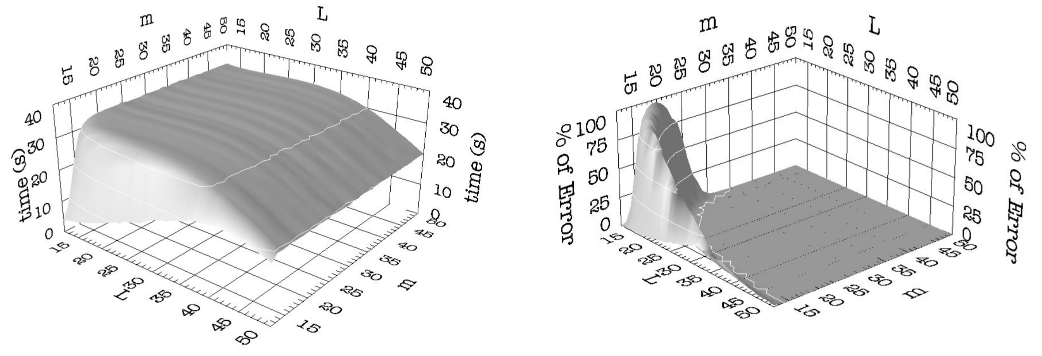
Finally, it is worth recalling the table method of Di Mauro et al. (2003) and the experimental results reported for it in Di Mauro et al. (2003) and Maloberti and Suzuki (2004) for simple subsumption as well as for enumerating all solutions. Both papers report that the table method does solve the challenge problems, so indeed it avoids some of the space problems incurred with our deterministic version. But both papers indicate that DJANGO is at least an order of magnitude faster and often much more, where the relation in run times depend on whether the problems are in the phase transition, over constrained region or under constrained region. These observations seem to agree with our comparisons of the table based and DJANGO methods.

To summarize, the subsumption experiments suggest that in general the DJANGO methods are more robust than the table based methods and that DJANGOPRIMAL is a promising alternative. However, these results must be interpreted with caution, since in these experiments the tables meth-



(A) (B)

Figure 4: (A) Percentage of wrong subsumption tests for randomized tables method on 50×50 pairs $(\mathcal{C}, Ex)$, with TH= 1 and R= 1. (B) Percentage of satisfiable subsumption tests. Notice that the table based method with such low valued parameters is not able to discover any subsumption substitution, and hence the error rate corresponds to the satisfiability rate of the subsumption problem suite.



(A) (B)

Figure 5: (A) Subsumption cost(m, L) and (B) error percentage for randomized tables method with TH= 10 and R= 10 averaged over 50×50 pairs $(\mathcal{C}, Ex)$.

ods look for all solutions while DJANGOPRIMAL and DJANGO only search for a single solution. Moreover, the settings chosen are probably better for the primal representation and other parameters, such as arity of literals, must also be investigated. In the context of LOGAN-H, the success of the table methods may be explained by the fact that the algorithm starts with long clauses and makes them shorter in the process. Thus it starts in the over constrained region and moves toward the phase transition and then possibly the under constrained region. Since tables perform well in the over constrained region and boundary to the phase transition they can do well overall. DJANGOPRIMAL was also faster in some of the experiments with LOGAN-H. Therefore, further optimization of DJANGOPRIMAL and adding randomized facilities to DJANGO are natural directions to improve the system.

6. Conclusion

The paper introduced the system LOGAN-H implementing new algorithms for learning function free Horn expressions. The system is based on algorithms proved correct in Khardon (1999b) but includes various improvements in terms of efficiency as well as a new batch algorithm that learns from examples only. The batch algorithm can be seen as performing a refinement search over multi-clause hypotheses. The main difference from other systems is that our algorithm is using a bottom up search and that it is using large refinement steps in this process. We demonstrated through qualitative and quantitative experiments that the system performs well in several benchmark ILP tasks. Thus our system gives competitive performance on a range of tasks while taking a completely different algorithmic approach, a property that is attractive when exploring new problems and applications.

The paper also introduced new algorithms for solving the subsumption problem and evaluated their performance. The table based methods give competitive performance within LOGAN-H and DJANGOPRIMAL is a promising new approach where hard subsumption problems in the phase transition region are solved.

As illustrated using the Bongard domain, LOGAN-H is particularly well suited to domains where sub-structures of examples in the data set are likely to be in the data set as well. On the other hand, for problems with a small number of examples where each example has a large number of objects and dramatically different structure our system is likely to overfit since there is little evidence for useful minimization steps. Indeed we found this to be the case for the the artificial challenge problems of Giordana et al. (2003) where our system outputs a large number of rules and gets low accuracy. This suggests that skipping the minimization step may lead to improved performance in such cases if pairings reduce clause size considerably. Initial experiments with this are as yet inconclusive.

Our system demonstrates that using large refinement steps with a bottom up search can be an effective inference method. As discussed above, bottom up search suffers from two aspects: subsumption tests are more costly than in top down approaches, and overfitting may occur in small data sets with large examples. On the other hand, it is not clear how large refinement steps or insights gained by using LGG can be used in a top down system. One interesting idea in this direction is given in the system of Bianchetti et al. (2002). Here repeated pairing-like operations are performed without evaluating the accuracy until a syntactic condition is met (this is specialized for the challenge problems of Giordana et al. (2003)) to produce a short clause. This clause is then used as a seed for a small step refinement search that evaluates clauses as usual. Finding similar ideas that work without using special properties of the domain is an interesting direction for future work.

Finally, it would be interesting to explore real world applications of the interactive algorithm. The work on the robot scientist project (King et al., 2004) provides one such possibility. This work uses a chemistry lab robot to perform experiments on assays to aid in the process of learning. The lab experiments are very similar to the membership queries used by our algorithm, however queries may be limited to chemicals that can be synthesized by the system. Thus such an application will require adapting our algorithm to use only appropriate queries.

Acknowledgments

This work was partly supported by EPSRC grant GR/M21409 in the UK and by NSF Grant IIS-0099446 in the USA. In addition, Roni Khardon was supported by a Research Semester Fellowship Award from Tufts University. Stephen Kitt at the University of Edinburgh and Constantine Ashminov at Tufts University made large contributions to the code in the system. We are grateful to Hendrik Blockeel who provided the Tilde system as well as a generator for Bongard examples, and Win Van Laer provided the ICL system. Mark Steedman made useful comments on earlier drafts. Some of the experiments were performed on a Linux cluster provided by Tufts Central Computing Services.

Appendix A. Further Details on Implementation and Applicability

The appendix gives additional details on important implementation issues leading to speedup and wider applicability of the system. In addition we give a discussion of using our learning from interpretations system on data given in the normal ILP setting.

A.1 Noisy Data Sets

Many real life data sets are "noisy", in the sense that there is no Horn expression hypothesis that is consistent with all the data. In this case the batch algorithm described above is not well defined since it assumes in step 2(b) that at least one potential consequent is correct and is therefore not removed in *one-pass*. However, if the data set is inconsistent it is possible that all potential consequents are removed. If this happens our system simply marks the counter-example as inconsistent and ignores it in future tests.

A related problem occurs when we use randomized subsumption tests. Here since the subsumption test is incomplete we may not notice that a rule in the hypothesis is violated by a negative example. As a result the algorithm may see the same negative counter-example multiple times. Note that since pairings move atoms from antecedent to conclusion the same counter-example may be encountered more than once even under normal conditions. To handle this, the system uses a small constant (5 in the experiments) to bound the number of times an example may be used as a counter-example. If this bound is exceeded the negative example is ignored in future tests.

A.2 Caching

The algorithms described above may produce repeated calls to *one-pass* with the same antecedent since pairings of one clause set with several others may result in the same clause set. Thus it makes sense to cache the results of *one-pass*. Notice that there is a tradeoff in the choice of what to cache. If we try to cache a universally quantified expression then matching it requires a subsumption test which is expensive. We therefore opted to cache a ground syntactic version of the clause. For both algorithms the system caches interpretations rather than clauses or clause sets (the s part of $[s,c]$). In fact, for the batch algorithm we only need to cache positive interpretations—if a clause set $[s,\emptyset]$

was returned by *one-pass* then s does not imply any of the possible consequents and therefore it is a positive interpretation. Thus any new call to *one-pass* with s can be skipped. To achieve fast caching while increasing the chances of cache hits, the system caches and compares a normalized representation of the interpretation by sorting predicate and atom names. This is matched with the fact that pairing keeps object names of existing clause sets in the hypothesis. Thus the same object names and ordering of these are likely to cause cache hits. Caching can reduce or increase run time of the system, depending on the data set, the cost for subsumption for examples in the data set, and the rate of cache hits.

A.3 Live Pairings

The system also includes an optimization that reduces the number of pairings that are tested without compromising correctness. Recall that the algorithm has to test all injective mappings between the domains of two interpretations. We say that a mapping is *live* if every paired 2-object appears in the extension of at least one atom in the antecedent of the pairing. One can show that if the target expression is range restricted (that is, all variables in the consequent appear in the antecedent) then testing live mappings is sufficient. For example, consider pairing the examples $[s_a, c_a] = [[p(1,2), p(2,3)], [q(2,2), q(3,1)]]$ and $[s_b, c_b] = [[p(a,b), p(b,c), p(a,d)], [q(d,d), q(c,a)]]$. Then the mapping $\{1/a, 2/b, 3/c\}$ gives the pairing $[[p(1,2), p(2,3)], [q(3,1)]]$ which is live. On the other hand the mapping $\{1/a, 2/d, 3/c\}$ gives the pairing $[[p(1,2)], [p(2,3), q(2,2), q(3,1)]]$ and when forcing range restricted form we get $[[p(1,2)], [q(2,2)]]$. This pairing is not live since $3/c$ does not appear in it. So, this pairing can be ignored. Technically, due to range restricted form, it is enough to check this only on the antecedent part. For efficiency, instead of generating all injective matchings and filtering irrelevant ones, one can first collect a set of potential matched objects-pairs and generate matchings from these.

A.4 Discretization of Real-Valued Arguments

The system includes a capability for handling numerical data by means of discretization. This is standard in machine learning systems but several aspects are peculiar to the relational setting and to bottom up learning.

Notice first that unlike attribute-value data there may be more than one occurrence of the same type of value in the same example, in different arguments of a predicate, different instances of the same predicate or different predicates. To capture this we first divide the numerical attributes into "logical groups" by annotating the data set. For example the rows of a chess board will belong to the same group regardless of the predicate and argument in which they appear. The system can then determine the threshold values and possibly the number of bins to divide values into. For example, discretizing the `logp` attribute in the Mutagenesis domain with 4 thresholds (5 ranges), a value between threshold 1 and threshold 2 will yield: `[logp(logp_val.02), logp_val>00(logp_val.02), logp_val>01(logp_val.02), logp_val<02(logp_val.02), logp_val<03(logp_val.02), ...]`. Notice that we are using both $\geq$ and $\leq$ predicates so that the hypothesis can encode intervals of values. Also, we use predicate names that explicitly encode thresholds numbers, for example, `logp_val>00` or `logp_val<02`.

Several approaches to discretization choosing the number of bins and the boundaries have been proposed in the literature (Fayyad and Irani, 1993; Dougherty et al., 1995; Blockeel and De Raedt, 1997). Our experiments use the following two approaches. The *equal frequency* approach requires the number of bins as input and it assigns the boundaries by giving each bin the same number of occurrences of values. To select the number of bins automatically one must use some version of cross validation. The *Information Gain* (IG) approach introduced by Fayyad and Irani (1993) uses the information gain criterion to split the range of values using a decision tree (Quinlan, 1993) and stops splitting using a minimum description length measure. For relational data the IG method must be refined since more than one value of the same type may appear in the same example. This was already addressed by Blockeel and De Raedt (1998) where values were weighted according to the number of times they appear in the examples. However, Blockeel and De Raedt (1998) set the number of bins by hand since IG provided too few regions. In our experiments we use the original criterion.

An interesting aspect arises when using discretization which highlights the way our system works and potential limitations. Recall that the system starts with an example and essentially turns objects into variables in the maximally specific clause set. It then evaluates this clause on other examples. Since we do not expect examples to be identical or very close, the above relies on the universal quantification to allow matching one structure into another. However, the effect of discretization is to ground the value of the discretized object. For example, if we discretized the `logp` attribute from above and variabilize we get `logp(X) logp_val>00(X) logp_val>01(X) logp_val<02(X) logp_val<03(X)`. Thus unless we drop some of the boundary constraints this limits matching examples to have a value in the same bin. We are therefore losing the power of universal quantification. As a result fewer positive examples will match in the early stages of the minimization process, less consequents will be removed, and the system may be led to overfitting by dropping the wrong objects. Thus while including all possible boundaries gives maximum flexibility in the hypothesis language this may not be desirable due to the danger of overfitting.

This point is illustrated by the following experiment discretizing values in the KRK domain. In this domain given an example's predicate `position(x1,x2,y1,y2,z1,z2)`, we consider the three values corresponding to columns `(x1,y1,z1)` as the same logical attribute and therefore we discretize them together. Similarly, we discretize the values of `(x2,y2,z2)` together. Versions of `adj()` for both column and row values are used. Since in this domain the number of bins and the boundaries for discretization are obvious we used the equal frequency method with the appropriate number of bins. As can be seen in Table 10 good accuracy is maintained with discretization but the learning curve is much slower than the non discretized version presented earlier. Another interesting point is that now "relevant background mode" performs much worse than "all background mode". In hindsight one can see that this is a result of the grounding effect of discretization. With "relevant background mode" the discretization threshold predicates and the adjacent predicates are different in every example. Since the examples are essentially ground we expect less matches between different examples and thus the system is likely to overfit. With "all background mode" these predicates do not constrain the matching of examples.

A.5 Pruning Rules

The system performs bottom up search and may stop with relatively long rules if the data is not sufficiently rich (that is, we do not have enough negative examples) to warrant further refinement of the rules. Pruning allows us to drop additional parts of rules. The system can perform a greedy reduced error pruning (Mitchell, 1997) using a validation data set. For each atom in the rule the system evaluates whether the removal of the atom increases the error on the validation set. If not,

	25	50	75	100	200	500	1000	2000	3000
Relevant background mode	43.32	43.70	45.05	44.60	52.39	72.26	84.80	90.30	92.17
All background mode	67.27	72.69	75.15	78.00	82.68	88.60	91.03	91.81	92.01

Table 10: Performance summary for KRK-illegal data set with discretization in average percentage accuracy.

the atom can be removed. While it is natural to allow an increase in error using a tradeoff against the length of the hypothesis in an MDL fashion, we have not yet experimented with this possibility.

Notice that unlike top down systems we can perform this pruning on the training set and do not necessarily need a separate validation set. In a top down system one grows the rules until they are consistent with the data. Thus, any pruning will lead to an increase in training set error. On the other hand in a bottom up system, pruning acts like the main stage of the algorithm in that it further generalizes the rules. In some sense, pruning on the training set allows us to move from a most specific hypothesis to a most general hypothesis that matches the data. Both training set pruning and validation set pruning are possible with our system.

A.6 Restricting Legal Conclusions

Our algorithm allows any predicate and instantiation to serve as a conclusion in a clause. However, when applying to a real data set we may know (or want to set up) that some predicates serve as rule conclusions and others do not. This is simply implemented in the *rel-cands* operation. This feature is useful both for applicability and in reducing run time that would be needed to generate, evaluate and remove rules with wrong conclusions.

A.7 Applicability to Data from Normal ILP Setting

In the normal ILP setting (Muggleton and DeRaedt, 1994) one is given a database as background knowledge and examples are simple atoms. We transform these into a set of interpretations as follows (see also De Raedt, 1997; Khardon, 1999b). The background knowledge in the normal ILP setting can be typically partitioned into different subsets such that each subset affects a single example only. A similar effect is achieved for intensional background knowledge in the Progol system (Muggleton, 1995) by using mode declarations to limit antecedent structure. Given example b, we will denote $BK(b)$ as the set of atoms in the background knowledge that is relevant to b. In the normal ILP setting we have to find a theory T s.t. $BK(b) \cup T \models b$ if b is a positive example, and $BK(b) \cup T \not\models b$ if b is negative. Equivalently, T must be such that $T \models BK(b) \rightarrow b$ if b is positive and $T \not\models BK(b) \rightarrow b$ if b is negative.

If b is a positive example in the standard ILP setting then we can construct an interpretation $I = ([V], [BK(b)])$ where V is the set of objects appearing in $BK(b)$, and label I as negative. When LOGAN-H finds the negative interpretation I, it constructs the set $[s, c] = \textit{rel-cands}(I)$ from it (notice that b is among the conclusions considered in this set), and then runs *one-pass* to figure out which consequents among the candidates are actually correct. Adding another interpretation $I' = ([V], [BK(b) \cup \{b\}])$ labeled positive guarantees that all other consequents are dropped. Notice that in order for this to be consistent with the target concept, we have to assume that the antecedent $BK(b)$ only implies b.

If b is a negative example in the standard ILP setting, we construct an interpretation $I = ([V],[BK(b)])$, where V is the set of variables appearing in $BK(b)$, and label it positive. Notice that if the system ever considers the clause $BK(b) \rightarrow b$ as a candidate, *one-pass* will find the positive interpretation I and will drop b, as desired. In fact, *one-pass* will return $[BK(b), \emptyset]$ for any input clause set $[BK(b), c]$ since it will drop all consequents in c that are not included in $BK(b)$ (including `false`), thus precluding clause $BK(b) \rightarrow b$ from ever being included in any hypothesis. This again assumes that no consequent can be implied by $BK(b)$.

Several ILP domains are formalized using a consequent of arity 1 where the argument is an object that identifies the example in the background knowledge. In this case, since we separate the examples into interpretations we do not need example identifiers and we get a consequent of arity 0. For learning with a single possible consequent of arity 0 our transformation can be simplified in that the extra positive example $I' = ([V],[BK(b) \cup \{b\}])$ is not needed since there are no other potential consequents. Thus we translate every positive example into a negative interpretation example and vice versa. As an example, suppose that in the normal ILP setting, the clause $p(a,b) \wedge p(b,c) \rightarrow q()$ is labeled positive and the clause $p(a,b) \rightarrow q()$ is labeled negative. Then, the transformed data set contains: $([a,b,c],[p(a,b),p(b,c)])-$ and $([a,b],[p(a,b)])+$. Notice that in this case the assumptions made regarding other consequents in the general transformation are not needed.

In the case of zero arity consequents, the check whether a given clause C is satisfied by some interpretation I can be considerably simplified. Instead of checking all substitutions it suffices to check for existence of some substitution, since any such substitution will remove the single nullary consequent. As a result subsumption procedures that enumerate solutions one by one can quit early, after the first substitution, and are likely to be faster in this case. In addition, note that the pairing operation never moves new atoms into the consequent and is therefore a pure generalization operation.

A.8 Algorithmic Details on DJANGO

This section provides some of the details on the algorithms used in DJANGOPRIMAL and DJANGODUAL and the difference between them. Both versions first build the domain of the variables applying a node-consistency procedure. Node-consistency removes from variable domains all values that do not satisfy unary constraints. Unary constraints are created when a variable has multiple occurrences in a literal.

Both methods continue by checking for arc-consistency. Arc consistency means that we remove from the domains all values which are not supported. A value is consistent with respect to a constraint, if there exists a value in the domains of all other variables of a constraint such that the constraint is satisfied. A value is supported if it is consistent for all constraints involving this variable. Obviously, a value not supported cannot be assigned to a variable, since a constraint will not be satisfied. DJANGODUAL uses a standard algorithm of arc-consistency named AC3 (Mackworth, 1977), which maintains a queue of all constraints that must be checked. The queue is initially full, and if a value is removed from a domain, all other constraints involving the variable are added to the queue. The algorithm checks and removes all elements in the queue until it is empty. AC3 is not optimal since it can check the same constraint more than once, however it has good average execution times. DJANGOPRIMAL uses a version of AC3 adapted to n-ary CSPs, named CN (Mackworth,

1977) or GAC3, for General Arc Consistency. Finally both methods perform a depth-first search using lookahead strategy; a partial solution is incrementally built assigning a value to one variable at a time. The variable is selected using Dynamic Variable Ordering (DVO); for each variable the size of its domain is divided by the number of constraints involving it, the variable with the minimal ratio is selected. This heuristic follows the First Fail Principle that tries to reduce the search tree by causing failure earlier during the search. All values inconsistent with this partial solution are removed from the domains of the variables not yet assigned. Three different strategies are used to propagate the effects of the last assignment:

Forward Checking (FC) only checks variables connected to the last assigned variable.

Maintaining Arc Consistency (MAC), also called Full Arc Consistency, extends FC by checking arc-consistency on adjacent variables if a value has been removed from their domain.

Forward Checking Singleton extends FC by also checking all constraints involving a domain containing a single value. Checking such constraint is cheaper than full arc consistency and can reach to a failure sooner than simple FC.

DJANGOPRIMAL uses FC, while DJANGODUAL uses a more sophisticated strategy, named Meta-DJANGO, which selects FC or MAC according to a measure, denoted κ (Gent et al., 1996), estimating the position of the problem relatively to the Phase Transition (Giordana and Saitta, 2000). Instances in the under-constrained region use FC, while instances in the phase transition and the over constrained regions use MAC.

In addition to the techniques above DJANGODUAL uses the idea of *signatures* to further prune the search. A signature captures the idea that when we map a literal in a clause onto a literal in the example all the neighborhood of the first literal must exist in the example as well. By encoding neighborhoods of literals which are Δ-variables in the dual representation we can prune the domain in similarity to arc-consistency. The details of this heuristic are given in Maloberti and Sebag (2004).

References

D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1988.

D. Angluin, M. Frazier, and L. Pitt. Learning conjunctions of Horn clauses. *Machine Learning*, 9: 147–164, 1992.

M. Arias and R. Khardon. Bottom-up ILP using large refinement steps. In *Proceeding of the International conference on Inductive Logic Programming*, pages 26–43. Springer, 2004. LNAI 3194.

M. Arias and R. Khardon. Learning closed Horn expressions. *Information and Computation*, 178: 214–240, 2002.

H. Arimura. Learning acyclic first-order Horn sentences from entailment. In *Proceedings of the International Conference on Algorithmic Learning Theory*, pages 432–445. Springer, 1997. LNAI 1316.

Jacques Alès Bianchetti, Céline Rouveirol, and Michèle Sebag. Constraint-based learning of long relational concepts. In *Proceedings of the International Conference on Machine Learning*, pages 35–42. Morgan Kaufmann, 2002.

H. Blockeel and L. De Raedt. Lookahead and discretization in ILP. In *Proceedings of the 7th International Workshop on Inductive Logic Programming*, pages 77–84. Springer, 1997. LNAI 1297.

H. Blockeel and L. De Raedt. Top down induction of first order logical decision trees. *Artificial Intelligence*, 101:285–297, 1998.

H Blockeel, L. De Raedt, N. Jacobs, and B. Demoen. Scaling up inductive logic programming by learning from interpretations. *Data Mining and Knowledge Discovery*, 3(1):59–93, 1999.

H. Blockeel, L. Dehaspe, B. Demoen, G. Janssens, J. Ramon, and H. Vandecasteele. Executing query packs in ILP. In *Proc. Tenth International Conference on Inductive Logic Programming*, LNAI 1866, pages 60–77. Springer-Verlag, Berlin, 2000.

H. Blockeel, L. Dehaspe, B. Demoen, G. Janssens, J. Ramon, and H. Vandecasteele. Improving the efficiency of inductive logic programming through the use of query packs. *Journal of Artificial Intelligence Research*, 16:135–166, 2002.

A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Occam's razor. *Information Processing Letters*, 24:377–380, April 1987.

I. Bratko. Refining complete hypotheses in ILP. In *International Workshop on Inductive Logic Programming*, pages 44–55. Springer, 1999. LNAI 1634.

I. Bratko and S. Muggleton. Applications of inductive logic programming. *Communications of the ACM*, 38(11):65–70, November 1995.

C. Chang and J. Keisler. *Model Theory*. Elsevier, Amsterdam, Holland, 1990.

X. Chen. *A Theoretical Comparison of Selected CSP Solving and Modeling Techniques*. PhD thesis, University of Alberta, Canada, 2000.

L. De Raedt. Logical settings for concept learning. *Artificial Intelligence*, 95(1):187–201, 1997. See also relevant Errata (forthcoming).

L. De Raedt and S. Dzeroski. First order *jk*-clausal theories are PAC-learnable. *Artificial Intelligence*, 70:375–392, 1994.

L. De Raedt and W. Van Laer. Inductive constraint logic. In *Proceedings of the 6th Conference on Algorithmic Learning Theory*. Springer, 1995. LNAI 997.

R. Dechter and J. Pearl. Structure identification in relational data. *Artificial Intelligence*, 58:237–270, 1992.

N. Di Mauro, T.M.A. Basile, S. Ferilli, F. Esposito, and N. Fanizzi. An exhaustive matching procedure for the improvement of learning efficiency. In *Proceedings of the 13th International Conference on Inductive Logic Programming*, pages 112–129. Springer, 2003. LNAI 2835.

J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *Proceedings of the International Conference on Machine Learning*, pages 194–202, 1995.

U.M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the International Conference on Machine Learning*, pages 1022–1027, 1993.

I.P. Gent, E. MacIntyre, P. Prosser, and T. Walsh. The constrainedness of search. In *Proceedings of the National Conference on Artificial Intelligence*, pages 246–252, 1996.

A. Giordana and L. Saitta. Phase transitions in relational learning. *Machine Learning*, 2:217–251, 2000.

A. Giordana, L. Saitta, M. Sebag, and M. Botta. Relational learning as search in a critical region. *Journal of Machine Learning Research*, 4:431–463, 2003.

H. Kautz, M. Kearns, and B. Selman. Horn approximations of empirical data. *Artificial Intelligence*, 74:129–145, 1995.

R. Khardon. Learning horn expressions with LogAn-H. In *Proceedings of the International Conference on Machine Learning*, pages 471–478. Morgan Kaufmann, 2000.

R. Khardon. Learning range-restricted Horn expressions. In *Proceedings of the Fourth European Conference on Computational Learning Theory*, pages 111–125. Springer, 1999a. LNAI 1572.

R. Khardon. Learning function free Horn expressions. *Machine Learning*, 37(3):249–275, 1999b.

R. D. King, K. Whelam, F. Jones, P. Reiser, C. Bryant, S. Muggleton, D. Kell, and S. Oliver. Functional genomic hypothesis generation and experimentation by a robot scientist. *Nature*, 427: 247–252, 2004.

M. Krishna Rao and A. Sattar. Learning from entailment of logic programs with local variables. In *Proceedings of the International Conference on Algorithmic Learning Theory*. Springer, 1998. LNAI 1501.

W. Van Laer, H. Blockeel, and L. De Raedt. Inductive constraint logic and the mutagenesis problem. In *Proceedings of the Eighth Dutch Conference on Artificial Intelligence*, pages 265–276, November 1996.

J.W. Lloyd. *Foundations of Logic Programming*. Springer Verlag, 1987. Second Edition.

A.K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8(1):99–118, 1977.

J. Maloberti and M. Sebag. Fast theta-subsumption with constraint satisfaction algorithms. *Machine Learning*, 55(2):137–174, 2004.

J. Maloberti and E. Suzuki. An efficient algorithm for reducing clauses based on constraint satisfaction techniques. In *Proceedings of the 14th International Conference on Inductive Logic Programming*, pages 234–251. Springer Verlag LNAI 3194, 2004.

T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

S. Muggleton. Inverse entailment and Progol. *New Generation Computing, Special issue on Inductive Logic Programming*, 13(3-4):245–286, 1995.

S. Muggleton and W. Buntine. Machine invention of first order predicates by inverting resolution. In S. Muggleton, editor, *Inductive Logic Programming*. Academic Press, 1992.

S. Muggleton and L. DeRaedt. Inductive logic programming: Theory and methods. *The Journal of Logic Programming*, 19 & 20:629–680, May 1994.

S. Muggleton and C. Feng. Efficient induction of logic programs. In S. Muggleton, editor, *Inductive Logic Programming*, pages 281–298. Academic Press, 1992.

S. H. Muggleton, M. Bain, J. Hayes-Michie, and D. Michie. An experimental comparison of human and machine learning formalisms. In *Proc. Sixth International Workshop on Machine Learning*, pages 113–118. Morgan Kaufmann, 1989.

F. Pereira and S. Shieber. *Prolog and natural-language analysis*. Stanford : Center for the Study of Language and Information, 1987.

G. D. Plotkin. A note on inductive generalization. *Machine Intelligence*, 5:153–163, 1970.

G. D. Plotkin. A further note on inductive generalization. *Machine Intelligence*, 6:101–124, 1971.

J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.

J. R. Quinlan. *C4.5: Programs for machine learning*. Morgan Kaufmann, 1993.

C. Reddy and P. Tadepalli. Learning first order acyclic Horn programs from entailment. In *International Conference on Inductive Logic Programming*, pages 23–37. Springer, 1998. LNAI 1446.

V. Santos Costa, A. Srinivasan, R. Camacho, H. Blockeel, B. Demoen, G. Janssens, J. Struyf, H. Vandecasteele, and W. Van Laer. Query transformations for improving the efficiency of ILP systems. *Journal of Machine Learning Research*, 4:465–491, 2003.

M. Sebag and C. Rouveirol. Resource-bounded relational reasoning: Induction and deduction through stochastic matching. *Machine Learning*, 38:41–62, 2000.

A. Srinivasan, S. H. Muggleton, R. D. King, and M. J. E. Sternberg. Mutagenesis: ILP experiments in a non-determinate biological domain. In S. Wrobel, editor, *Proc. 4th Int. Workshop on Inductive Logic Programming*, pages 217–232, September 1994.

A. Srinivasan, S. Muggleton, and R.D. King. Comparing the use of background knowledge by inductive logic programming systems. In *Proceedings of the 5th International Workshop on Inductive Logic Programming*, pages 199–230, 1995.

C. Stolle, A. Karwath, and L. De Raedt. CLASSIC'CL: An integrated ILP system. In *Discovery Science*, pages 354–362, 2005.

E. Tsang. *Foundations of Constraint Satisfaction*. Academic Press, London, 1993.

Journal of Machine Learning Research 8 (2007) 589-612 Submitted 5/06; Revised 10/06; Published 3/07

Consistent Feature Selection for Pattern Recognition in Polynomial Time

Roland Nilsson ROLLE@IFM.LIU.SE
José M. Peña JMP@IFM.LIU.SE
Division of Computational Biology, Department of Physics, Chemistry and Biology
The Institute of Technology
Linköping University
SE-581 83 Linköping, Sweden

Johan Björkegren JOHAN.BJORKEGREN@KI.SE
The Computational Medicine Group, Center for Molecular Medicine
Department of Medicine
Karolinska Institutet, Karolinska University Hospital Solna
SE-171 76 Stockholm, Sweden

Jesper Tegnér JESPERT@IFM.LIU.SE
Division of Computational Biology, Department of Physics, Chemistry and Biology
The Institute of Technology
Linköping University
SE-581 83 Linköping, Sweden

Editor: Leon Bottou

Abstract

We analyze two different feature selection problems: finding a minimal feature set optimal for classification (MINIMAL-OPTIMAL) *vs.* finding all features relevant to the target variable (ALL-RELEVANT). The latter problem is motivated by recent applications within bioinformatics, particularly gene expression analysis. For both problems, we identify classes of data distributions for which there exist consistent, polynomial-time algorithms. We also prove that ALL-RELEVANT is much harder than MINIMAL-OPTIMAL and propose two consistent, polynomial-time algorithms. We argue that the distribution classes considered are reasonable in many practical cases, so that our results simplify feature selection in a wide range of machine learning tasks.

Keywords: learning theory, relevance, classification, Markov blanket, bioinformatics

1. Introduction

Feature selection (FS) is the process of reducing input data dimension. By reducing dimensionality, FS attempts to solve two important problems: facilitate learning (inducing) accurate classifiers, and discover the most "interesting" features, which may provide for better understanding of the problem itself (Guyon and Elisseeff, 2003).

The first problem has been extensively studied in pattern recognition research. More specifically, the objective here is to learn an optimal classifier using a minimal number of features; we refer to this as the MINIMAL-OPTIMAL problem. Unfortunately, MINIMAL-OPTIMAL is in general intractable even asymptotically (in the large-sample limit), since there exist data distributions for which every feature subset must be tested to guarantee optimality (Cover and van Campenhout, 1977). Therefore it is common to resort to suboptimal methods. In this paper, we take a different approach to solving MINIMAL-OPTIMAL: we restrict the problem to the class of *strictly positive* data distributions, and prove that within this class, the problem is in fact polynomial in the number of features. In particular, we prove that a simple backward-elimination algorithm is asymptotically optimal. We then demonstrate that due to measurement noise, most data distributions encountered in practical applications are strictly positive, so that our result is widely applicable.

The second problem is less well known, but has recently received much interest in the bioinformatics field, for example in gene expression analysis (Golub et al., 1999). As we will explain in Section 4, researchers in this field are primarily interested in identifying *all* features (genes) that are somehow related to the target variable, which may be a biological state such as "healthy" *vs.* "diseased" (Slonim, 2002; Golub et al., 1999). This defines the ALL-RELEVANT problem. We prove that this problem is much harder than MINIMAL-OPTIMAL; it is asymptotically intractable even for strictly positive distributions. We therefore consider a more restricted but still reasonable data distribution class, and propose two polynomial algorithms for ALL-RELEVANT which we prove to be asymptotically optimal within that class.

2. Preliminaries

In this section we review some concepts needed for our later developments. Throughout, we will assume a binary classification model where training examples $(x^{(i)}, y^{(i)}), i = 1, \dots, l$ are independent samples from the random variables (X,Y) with density $f(x,y)$, where $X = (X_1, \dots, X_n) \in \mathbb{R}^n$ is a sample vector and $Y \in \{-1,+1\}$ is a sample label. Capital X_i denote random variables, while lowercase symbols x_i denote observations. We will often treat the data vector X as a set of variables, and use the notation $R_i = X \setminus \{X_i\}$ for the set of all features except X_i. Domains of variables are denoted by calligraphic symbols $\mathcal{X}$. We will present the theory for continuous X, but all results are straightforward to adapt to the discrete case. Probability density functions (continuous variables) or probability mass functions (discrete variables) are denoted $f(x)$ and $p(x)$, respectively. Probability of events are denotes by capital P, for example, $P(Y = 1/2)$.

2.1 Distribution Classes

In the typical approach to FS, one attempts to find heuristic, suboptimal solutions while considering all possible data distributions $f(x,y)$. In contrast, we will restrict the feature selection problem to certain classes of data distributions in order to obtain optimal solutions. Throughout, we will limit ourselves to the following class.

Definition 1 *The class of strictly positive data distributions consists of the $f(x,y)$ that satisfies (i) $f(x) > 0$ almost everywhere (in the Lebesgue measure) and (ii) $P(p(y|X) = 1/2) = 0$.*

Note we do not require $f(x,y) > 0$, which would be more restrictive. The criterion (i) states that a set in $\mathcal{X}$ has nonzero probability iff it has nonzero Lebesgue measure, while (ii) states that the optimal decision boundary has zero measure. These conditions are mainly needed to ensure uniqueness of the Bayes classifier (see below). It is reasonable to assume that this class covers the great majority of practical pattern recognition problems, since most data originates in physical measurements of some kind and is inevitably corrupted by noise. For example, consider the additive Gaussian noise

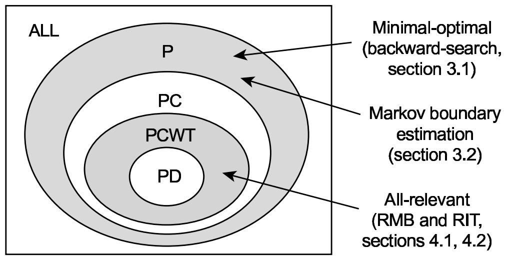

ALL
P
PC
PCWT
PD
Minimal-optimal (backward-search, section 3.1)
Markov boundary estimation (section 3.2)
All-relevant (RMB and RIT, sections 4.1, 4.2)

Figure 1: The distribution classes used. P, strictly positive; PC, strictly positive satisfying the composition property; PCWT, strictly positive satisfying composition and weak transitivity; PD, strictly positive and DAG-faithful. Arrows show classes where the various algorithms (right) are proved to be consistent.

model

$$X = x_0 + \varepsilon, \quad \varepsilon \sim N(0, \sigma).$$

Since the noise component ε is strictly positive over the domain of X, we immediately obtain $f(x) > 0$. A similar argument holds for binary data with Bernoulli noise, and indeed for any additive noise model with $f(\varepsilon) > 0$. In general, the strictly positive restriction is considered reasonable whenever there is uncertainty about the data (Pearl, 1988). Note that the $f(x) > 0$ criterion by definition only concerns the actual domain $\mathcal{X}$. If the data distribution is known to be constrained for physical reasons to some compact set such as $0 < X < 1$, then naturally $f(x) > 0$ need not hold outside that set. Typical problems violating $f(x) > 0$ involve noise-free data, such as inference of logic propositions (Valiant, 1984).

In Section 4, we will consider the more narrow classes of strictly positive distributions (P) that also satisfy the *composition* property (PC), and those in PC that additionally satisfies the *weak transitivity* property (PCWT). See Appendix B for details on these properties. Although these restrictions are more severe than $f(x) > 0$, these classes still allow for many realistic models. For example, the jointly Gaussian distributions are known to be PCWT (Studený, 2004). Also, the strictly positive and *DAG-faithful* distributions (PD) are contained in PCWT (Theorem 24, Appendix B). However, we hold that the PCWT class is more realistic than PD, since PCWT distributions will remain PCWT when a subset of the features are marginalized out (that is, are not observed), while PD distributions may not (Chickering and Meek, 2002).[1] This is an important argument in favor of PCWT, as in many practical cases we cannot possibly measure all variables. An important example of this is gene expression data, which is commonly modelled by PD distributions (Friedman, 2004). However, it is frequently the case that not all genes can be measured, so that PCWT is a more realistic model class. Of course, these arguments also apply to the larger PC class. Figure 1 summarizes the relations between the distribution classes discussed in this section.

1. The paper by Chickering and Meek (2002) proves that the composition property is preserved under marginalization. A similar proof for the weak transitivity property can be found in Peña et al. (2006).

2.2 Classifiers, Risk and Optimality

A *classifier* is defined as a function $g(x) : \mathcal{X} \mapsto \mathcal{Y}$, predicting a category y for each observed example x. The "goodness" of a classifier is measured as follows.

Definition 2 (risk) *The risk $R(g)$ of a classifier g is*

$$R(g) = P(g(X) \neq Y) = \sum_{y \in \mathcal{Y}} p(y) \int_{\mathcal{X}} 1_{\{g(x) \neq y\}} f(x|y) dx, \tag{1}$$

where $1_{\{\cdot\}}$ is the set indicator function.

For a given distribution $f(x,y)$, an (optimal) *Bayes classifier* is one that minimizes $R(g)$. It is easy to show that the classifier g^* that maximizes the posterior probability,

$$g^*(x) = \begin{cases} +1, & P(Y = 1|x) \geq 1/2 \\ -1, & \text{otherwise} \end{cases}, \tag{2}$$

is optimal (Devroye et al., 1996). For strictly positive distributions, $g^*(x)$ is also unique, except on zero-measure subsets of $\mathcal{X}$ (above, the arbitrary choice of $g^*(x) = +1$ at the decision boundary $p(y|x) = 1/2$ is such a zero-measure set). This uniqueness is important for our results, so for completeness we provide a proof in Appendix A (Lemma 19). From now on, we speak of *the* optimal classifier g^* for a given f.

2.3 Feature Relevance Measures

Much of the theory of feature selection is centered around various definitions of feature *relevance*. Unfortunately, many authors use the term "relevant" casually and without a clear definition, which has caused much confusion on this topic. Defining relevance is not trivial, and there are many proposed definitions capturing different aspects of the concept; see for example Bell and Wang (2000) for a recent survey. The definitions considered in this paper are rooted in the well-known concept of (probabilistic) conditional independence (Pearl, 1988, sec. 2.1).

Definition 3 (conditional independence) *A variable X_i is conditionally independent of a variable Y given (conditioned on) the set of variables $S \subset X$ iff it holds that*

$$P\big(p(Y|X_i, S) = p(Y|S)\big) = 1.$$

This is denoted $Y \perp X_i | S$.

In the above, the $P(\dots) = 1$ is a technical requirement allowing us to ignore pathological cases where the posterior differs on zero-measure sets. Conditional independence is a measure of *irrelevance*, but it is difficult to use as an operational definition since this measure depends on the conditioning set S. For example, a given feature X_i can be conditionally independent of Y given $S = \emptyset$ (referred to as *marginal* independence), but still be dependent for some $S \neq \emptyset$. The well-known XOR problem is an example of this. Two well-known relevance definitions coping with this problem were proposed by John et al. (1994).

Definition 4 (strong and weak relevance) *A feature X_i is strongly relevant to Y iff $Y \not\perp X_i | R_i$. A feature X_i is weakly relevant to Y iff it is not strongly relevant, but satisfies $Y \not\perp X_i | S$ for some set $S \subset R_i$.*

Informally, a strongly relevant feature carries information about Y that cannot be obtained from any other feature. A weakly relevant feature also carries information about Y, but this information is "redundant"—it can also be obtained from other features. Using these definitions, relevance and irrelevance of a feature to a target variable Y are defined as

Definition 5 (relevance) *A feature X_i is relevant to Y iff it is strongly relevant or weakly relevant to Y. A feature X_i is irrelevant to Y iff it is not relevant to Y.*

Finally, we will need the following definition of relevance with respect to a classifier.

Definition 6 *A feature X_i is relevant to a classifier g iff*

$$P(g(X_i, R_i) \neq g(X_i', R_i)) > 0,$$

where X_i, X_i' are independent and identically distributed.

This definition states that in order to be considered "relevant" to g, a feature X must influence the value of $g(x)$ with non-zero probability. It is a probabilistic version of that given by Blum and Langley (1997). Note that here, X_i and X_i' are independent samplings of the feature X_i; hence their distributions are identical and determined by the data distribution $f(x,y)$. In the next section, we examine the relation between this concept and the relevance measures in Definition 4.

3. The Minimal-Optimal Problem

In practise, the data distribution is of course unknown, and a classifier must be induced from the training data $D^l = \{(x^{(i)}, y^{(i)})\}_{i=1}^l$ by an *inducer*, defined as a function $I : (\mathcal{X} \times \mathcal{Y})^l \mapsto \mathcal{G}$, where $\mathcal{G}$ is some space of functions. We say that an inducer is *consistent* if the induced classifier $I(D^l)$ converges in probability to g^* as the sample size l tends to infinity,

$$I(D^l) \xrightarrow{P} g^*.$$

Consistency is a reasonable necessary criterion for a sound inducer, and has been verified for a wide variety of algorithms (Devroye et al., 1996). Provided that the inducer used is consistent, we can address the feature selection problem asymptotically by studying the Bayes classifier. We therefore define the optimal feature set as follows.

Definition 7 *The* MINIMAL-OPTIMAL *feature set S^* is defined as the set of features relevant to the Bayes classifier g^* (in the sense of Definition 6).*

Clearly, S^* depends only on the data distribution, and is the minimal feature set that allows for optimal classification; hence its name. Since g^* is unique for strictly positive distributions (Lemma 19), it follows directly from Definition 6 that S^* is then also unique. Our first theorem provides an important link between the MINIMAL-OPTIMAL set and the concept of strong relevance.

Theorem 8 *For any strictly positive distribution $f(x,y)$, the* MINIMAL-OPTIMAL *set S^* contains only strongly relevant features.*

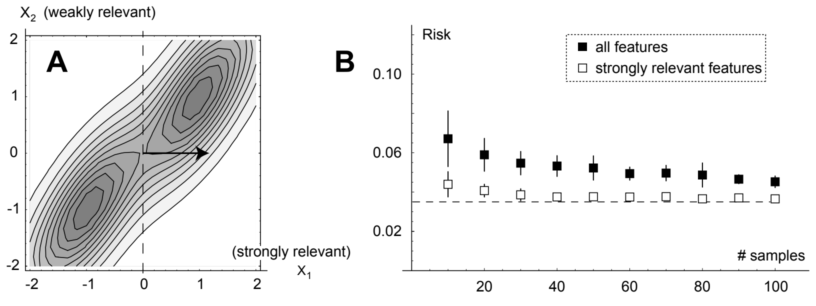


Figure 2: **A:** The example density $f(x_{2i-1}, x_{2i})$ given by (4). Here, X_1 is strongly relevant and X_2 weakly relevant. Arrow and dashed line indicates the optimal separating hyperplane. **B:** The risk functional $R(g)$ for a linear SVM trained on all relevant features (filled boxes) *vs.* on strongly relevant features only (open boxes), for the 10-dimensional distribution (4). Average and standard deviation over 20 runs are plotted against increasing sample size. The Bayes risk (dashed line) is $R(g^*) = 0.035$.

Proof Since f is strictly positive, g^* is unique by Lemma 19 (Appendix A). Consider any feature X_i relevant to g^*, so that $P(g^*(X_i, R_i) \neq g^*(X_i', R_i)) > 0$ by Definition 6. From the form (2) of the Bayes classifier, we find that

$$g^*(x_i, r_i) \neq g^*(x_i', r_i) \Rightarrow p(y|x_i, r_i) \neq p(y|x_i', r_i) \tag{3}$$

everywhere except possibly on the decision surface $\{x : p(y|x) = 1/2\}$. But this set has zero probability due to assumption (ii) of Definition 1. Therefore,

$$P(p(Y|X_i, R_i) \neq p(Y|X_i', R_i)) \geq P(g^*(X_i, R_i) \neq g^*(X_i', R_i)) > 0.$$

By Lemma 20 this is equivalent to $P(p(Y|X_i, R_i) = p(Y|R_i)) < 1$, which is the same as $Y \not\perp X_i | R_i$. Hence, X_i is strongly relevant. ■

Note that uniqueness of g^* is required here: if there would exist a different Bayes classifier g', the implication (3) would not hold. Theorem 8 is important because it asserts that we may safely ignore weakly relevant features, conditioned on the assumption $f(x) > 0$. This leads to more efficient (polynomial-time) algorithms for finding S^* for such problems. We will explore this consequence in Section 3.3.

An example illustrating Theorem 8 is given in Figure 2. Here, f is a 10-dimensional Gaussian mixture

$$f(x_1, \ldots, x_{10}, y) \propto \prod_{i=1}^{5} e^{-\frac{9}{8}((x_{2i-1}-y)^2 + (x_{2i-1}-x_{2i})^2)}. \tag{4}$$

Figure 2A shows the joint distribution of (X_{2i-1}, X_{2i}) (all such pairs are identically distributed). Note that, although the shape of the distribution in Figure 2 suggests that both features are relevant to Y, it is easy to verify directly from (4) that $X_2, X_4, \ldots, X_{10}$ are weakly relevant: considering for example the pair (X_1, X_2), we have

$$\begin{aligned} p(y|x_1,x_2) &= \frac{f(x_1,x_2,y)}{f(x_1,x_2)} \\ &= \left[1+\exp\left\{-\frac{9}{8}((x_1+y)^2-(x_1-y)^2)\right\}\right]^{-1} \\ &= \left[1+\exp\left\{-\frac{9x_1y}{2}\right\}\right]^{-1} \end{aligned}$$

which depends only on x_1, so X_2 is weakly relevant. The Bayes classifier is easy to derive from the condition $p(y|x) > 1/2$ (Equation 2) and turns out to be $g^*(x) = \operatorname{sgn}(x_1 + x_3 + x_5 + x_7 + x_9)$, so that $S^* = \{1,3,5,7,9\}$ as expected.

For any consistent inducer I, Theorem 8 can be treated as an approximation for finite (but sufficiently large) samples. If this approximation is fair, we expect that adding weakly relevant features will degrade the performance of I, since the Bayes risk must be constant while the *design cost* must increase (Jain and Waller, 1978). To illustrate this, we chose I to be a linear, soft-margin support vector machine (SVM) (Cortes and Vapnik, 1995) and induced SVM classifiers from training data sampled from the density (4), with sample sizes $l = 10, 20, \ldots, 100$. Figure 2B shows the risk of $g = I(D^l)$ and $g_{S^*} = I_{S^*}(D^l_{S^*})$ (here and in what follows we take g_{S^*}, I_{S^*}, and $D^l_{S^*}$ to mean classifiers/inducers/data using only the features in S^*). The SVM regularization parameter C was chosen by optimization over a range $10^{-2}, \ldots, 10^2$; in each case, the optimal value was found to be 10^2. We found that I_{S^*} does outperform I, as expected. The risk functional $R(g)$ was calculated by numerical integration of Equation (1) for each SVM hyperplane g and averaged over 20 training data sets. Clearly, adding the weakly relevant features increases risk in this example.

As the following example illustrates, the converse of Theorem 8 is false: there exist strictly positive distributions where even strongly relevant features are not relevant to the Bayes classifier.

Example 1 *Let $\mathcal{X} = [0,1]$, $\mathcal{Y} = \{-1,+1\}$, $f(x) > 0$ and $p(y{=}1|x) = x/2$. Here X is clearly strongly relevant. Yet, X is not relevant to the Bayes classifier, since we have $p(y=1|x) < 1/2$ almost everywhere (except at $x = 1$). We find that $g^*(x) = -1$ and $R(g^*) = P(Y{=}1)$.*

Clearly, this situation occurs whenever a strongly relevant feature X_i affects the value of the posterior $p(y|x)$ but not the Bayes classifier g^* (because the change in $p(y|x)$ is not large enough to alter the decision of $g^*(x)$). In this sense, relevance to the Bayes classifier is *stronger* than strong relevance.

3.1 Related Work

The relevance concepts treated above have been studied by several authors. In particular, the relation between the optimal feature set and strong *vs.* weak relevance was treated in the pioneering study by Kohavi and John (1997), who concluded from motivating examples that ”(i) all strongly relevant features and (ii) some of the weakly relevant ones are needed by the Bayes classifier”. As we have seen in Example 1, part (i) of this statement is not correct in general. Part (ii) is true in general, but Theorem 8 shows that this is not the case for the class of strictly positive f, and therefore it is rarely true in practise.

A recent study by Yu and Liu (2004) examines the role of weakly relevant features in more detail and subdivides these further into *redundant* and *non-redundant* weakly relevant features, of which the latter is deemed to be important for the Bayes classifier. However, Yu & Liu consider arbitrary distributions; for strictly positive distributions however, it is easy to see that all weakly relevant features are also ”redundant” in their terminology, so that their distinction is not useful in this case.

3.2 Connections with Soft Classification

A result similar to Theorem 8 have recently been obtained for the case of *soft* (probabilistic) classification (Hardin et al., 2004; Tsamardinos and Aliferis, 2003). In soft classification, the objective is to learn the posterior $p(y|x)$ instead of $g^*(x)$. By Definition 6, the features relevant to the optimal soft classifier $p(y|x)$ satisfies

$$P(p(Y|X_i,R_i) \neq p(Y|X_i',R_i)) > 0$$

which is equivalent to $P(p(Y|X_i,R_i) \neq p(Y|R_i)) > 0$ by Lemma 20. Thus, the features relevant to $p(y|x)$ are exactly the strongly relevant ones, so the situation in Example 1 does not occur here.

When learning soft classifiers from data, a feature set commonly encountered is the *Markov boundary* of the class variable, defined as the minimal feature set required to predict the posterior. Intuitively, this is the soft classification analogue of MINIMAL-OPTIMAL. The following theorem given by Pearl (1988, pp. 97) shows that this set is well-defined for strictly positive distributions.[2]

Theorem 9 (Markov boundary) *For any strictly positive distribution $f(x,y)$, there exists a unique minimal set $M \subseteq X$ satisfying $Y \perp X \setminus M|M$. This minimal set is called the Markov boundary of the variable Y (with respect to X) and denoted $M(Y|X)$.*

Tsamardinos & Aliferis recently proved that for the PD distribution class (see Figure 1), the Markov boundary coincides with the set of strongly relevant features (Tsamardinos and Aliferis, 2003). However, as explained in Section 2.1, the PD class is too narrow for many practical applications. Below, we generalize their result to any positive distribution to make it more generally applicable.

Theorem 10 *For any strictly positive distribution $f(x,y)$, a feature X_i is strongly relevant if and only if it is in the Markov boundary $M = M(Y|X)$ of Y.*

Proof First, assume that X_i is strongly relevant. Then $Y \not\perp X_i|R_i$, which implies $M \not\subseteq R_i$, so $X_i \in M$. Conversely, fix any $X_i \in M$ and let $M' = M \setminus \{X_i\}$. If X_i is not strongly relevant, then $Y \perp X_i|R_i$, and by the definition of the Markov boundary, $Y \perp X \setminus M|M$. We may rewrite this as

$$\begin{cases} Y \perp X_i|M' \cup X \setminus M \\ Y \perp X \setminus M|M' \cup \{X_i\}. \end{cases}$$

The intersection property (Theorem 25, Appendix B) now implies $Y \perp X \setminus M'|M'$. Hence, M' is a Markov blanket smaller than M, a contradiction. We conclude that X_i is strongly relevant. ■

The feature set relations established at this point for strictly positive distributions are summarized in Figure 3.

2. Pearl (1988) gives a proof assuming $f(x,y) > 0$. However, it is straightforward to relax this assumption to $f(x) > 0$ in our case, since we only consider Markov boundaries of Y. See Theorem 25 and corollary 26 in Appendix B.

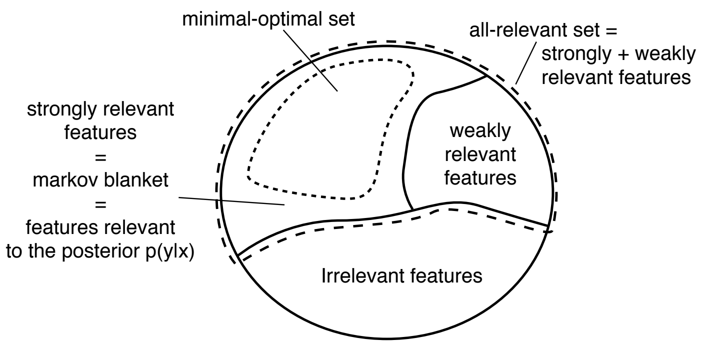


Figure 3: The identified relations between feature sets for strictly positive distributions. The circle represents all features. The dotted line (MINIMAL-OPTIMAL) denotes a subset, while the solid lines denote a partition into disjoint sets.

3.3 Consistent Polynomial Algorithms

By limiting the class of distributions, we have simplified the problem to the extent that weakly relevant features can be safely ignored. In this section, we show that this simplification leads to polynomial-time feature selection (FS) algorithms. A FS algorithm can be viewed as a function $\Phi(D^l) : (\mathcal{X} \times \mathcal{Y})^l \mapsto 2^X$, where 2^X denotes the power-set of X. For finite samples, the optimal Φ depends on the unknown data distribution f and the inducer I (Tsamardinos and Aliferis, 2003). Asymptotically however, we may use consistency as a reasonable necessary criterion for a "correct" algorithm. In analogue with consistency of inducers, we define a FS algorithm $\Phi(D^l)$ to be consistent if it converges in probability to the MINIMAL-OPTIMAL set,

$$\Phi(D^l) \xrightarrow{P} S^*.$$

Conveniently, consistency of Φ depends only on the data distribution f. Next, we propose a polynomial-time FS algorithm and show that it is consistent for any strictly positive f. As before, feature sets used as subscripts denote quantities using only those features.

Theorem 11 *Take any strictly positive distribution $f(x,y)$ and let $\hat{c}(D^l_S)$ be a real-valued criterion function such that, for every feature subset S,*

$$\hat{c}(D^l_S) \xrightarrow{P} c(S), \tag{5}$$

where $c(S)$ depends only on the distribution $f(x,y)$ and satisfies

$$c(S) < c(S') \iff R(g^*_S) < R(g^*_{S'}). \tag{6}$$

Then the feature selection method

$$\Phi(D^l) = \{i : \hat{c}(D^l_{R_i}) > \hat{c}(D^l) + \varepsilon\}$$

where $\varepsilon \in (0, \eta)$ with $\eta = \min_{i \in S^*}(c(R_i) - c(X))$, is consistent.

Proof Since f is strictly positive, S^* is unique by Lemma 19. By Definition 7 and the assumption (6) it holds that $X_i \in S^*$ iff $c(X) < c(R_i)$. First consider the case $X_i \in S^*$. Fix an $\varepsilon \in (0,\eta)$ and let $\varepsilon' = \min\{(\eta-\varepsilon)/2, \varepsilon/2\}$. Choose any $\delta > 0$. By (5) there exist an l_0 such that for all $l > l_0$,

$$P\left(\max_S |\hat{c}(D^l_S) - c(S)| > \varepsilon'\right) \leq \delta/2n$$

Note that since the power-set 2^X is finite, taking the maxima above is always possible even though (5) requires only point-wise convergence for each S. Therefore the events (i) $\hat{c}(D^l_X) < c(X) + \varepsilon'$ and (ii) $\hat{c}(D^l_{R_i}) > c(R_i) - \varepsilon'$ both have probability at least $1 - \delta/2n$. Subtracting the inequality (i) from (ii) yields

$$\begin{aligned} \hat{c}(D^l_{R_i}) - \hat{c}(D^l_X) &> c(R_i) - c(X) - 2\varepsilon' \\ &\geq c(R_i) - c(X) - (\eta - \varepsilon) \geq \varepsilon. \end{aligned}$$

Thus, for every $l > l_0$,

$$\begin{aligned} P(X_i \in \Phi(D^l)) &= P(\hat{c}(D^l_{R_i}) - \hat{c}(D^l_X) > \varepsilon) \\ &\geq P\left(\hat{c}(D^l_X) < c(X) + \varepsilon' \;\wedge\; \hat{c}(D^l_{R_i}) > c(R_i) - \varepsilon'\right) \\ &\geq P\left(\hat{c}(D^l_X) < c(X) + \varepsilon'\right) + P\left(\hat{c}(D^l_{R_i}) > c(R_i) - \varepsilon'\right) - 1 \\ &\geq 1 - \delta/n. \end{aligned}$$

For the converse case $X_i \notin S^*$, note that since $c(X) = c(R_i)$,

$$\begin{aligned} P(X_i \in \Phi(D^l)) &= P\left(\hat{c}(D^l_{R_i}) - \hat{c}(D^l_X) > \varepsilon\right) \\ &\leq P(|\hat{c}(D^l_{R_i}) - c(R_i)| + |c(X) - \hat{c}(D^l_X)| > \varepsilon) \\ &\leq P\left(|\hat{c}(D^l_{R_i}) - c(R_i)| > \frac{\varepsilon}{2} \;\vee\; |c(X) - \hat{c}(D^l_X)| > \frac{\varepsilon}{2}\right) \\ &\leq P\left(|\hat{c}(D^l_{R_i}) - c(R_i)| > \varepsilon'\right) + P\left(|c(X) - \hat{c}(D^l_X)| > \varepsilon'\right) \leq \delta/n \end{aligned}$$

where in the last line we have used $\varepsilon' \leq \varepsilon/2$. Putting the pieces together, we obtain

$$\begin{aligned} P(\Phi(D^l) = S^*) &= P(\Phi(D^l) \supseteq S^* \;\wedge\; \Phi(D^l) \subseteq S^*) \\ &= P(\forall i \in S^* : X_i \in \Phi(D^l) \;\wedge\; \forall i \notin S^* : X_i \notin \Phi(D^l)) \\ &\geq |S^*|(1-\delta/n) + (n - |S^*|)(1-\delta/n) - (n-1) = 1 - \delta. \end{aligned}$$

Since δ was arbitrary, the required convergence follows. ■

The requirement to choose an $\varepsilon < \eta$ may seem problematic, since in practise η depends on the true distribution $f(x,y)$ and hence is unobservable. For convergence purposes, this can be remedied by choosing a sequence $\varepsilon = \varepsilon(l) \to 0$, so that $\varepsilon < \eta$ will become satisfied eventually. In practise, the parameter ε controls the trade-off between precision and recall; a small ε gives high recall but low precision, and vice versa. If it is possible to estimate the distribution of $\hat{c}$, one might attempt to choose ε so as to control precision as recall as desired. However, this is a difficult issue where further research is necessary.

The algorithm Φ evaluates the criterion $\hat{c}$ precisely n times, so it is clearly polynomial in n provided that $\hat{c}$ is. The theorem applies to both filter and wrapper methods, which differ only in the choice of $\hat{c}(D_S^l)$ (Kohavi and John, 1997). To apply the theorem in a particular case, we need only verify that the requirements (5) and (6) hold. For example, let I be the k-NN rule with training data $D^{l/2} = \{(X_1,Y_1),\dots,(X_{l/2},Y_{l/2})\}$ and let $\hat{R}$ be the usual empirical risk estimate on the remaining samples $\{(X_{l/2+1},Y_{l/2+1}),\dots,(X_l,Y_l)\}$. Provided k is properly chosen, this inducer is known to be universally consistent,

$$P(R(I_S(D_S^{l/2})) - R(g_S^*) > \varepsilon) \le 2e^{-l\varepsilon^2/(144\gamma_S^2)}$$

where γ_S depends on $|S|$ but not on l (Devroye et al., 1996, pp. 170). Next, with a test set of size $l/2$, the empirical risk estimate satisfies

$$\forall g : P(|\hat{R}(g) - R(g)| > \varepsilon) \le 2e^{-l\varepsilon^2}$$

(Devroye et al., 1996, pp. 123). We choose $\hat{c}(D_S^l) = \hat{R}(I_S(D_S^{l/2}))$ and $c(S) = R(g_S^*)$, so that (6) is immediate. Further, this $\hat{c}(D_S^l)$ satisfies

$$\begin{aligned}
&P\left(|\hat{c}(D_S^l) - c(S)| > \varepsilon\right) = P\left(|\hat{R}(I_S(D_S^{l/2})) - R(g_S^*)| > \varepsilon\right)\\
&\le P\left(|\hat{R}(I_S(D_S^{l/2})) - R(I_S(D_S^{l/2}))| + |R(I_S(D_S^{l/2})) - R(g_S^*)| > \varepsilon\right)\\
&\le P\left(|\hat{R}(I_S(D_S^{l/2})) - R(I_S(D_S^{l/2}))| > \frac{\varepsilon}{2}\right) + P\left(|R(I_S(D_S^{l/2})) - R(g_S^*)| > \frac{\varepsilon}{2}\right)\\
&\le 2e^{-l\varepsilon^2/4} + 2e^{-l\varepsilon^2/(576\gamma_S^2)} \to 0
\end{aligned}$$

for every S as required by (5), and is polynomial in n. Therefore this choice defines a polynomial-time, consistent wrapper algorithm Φ. Note that we need only verify the point-wise convergence (5) for any given $\hat{c}(S)$, which makes the application of the theorem somewhat easier. Similarly, other consistent inducers and consistent risk estimators could be used, for example support vector machines (Steinwart, 2002) and the cross-validation error estimate (Devroye et al., 1996, chap. 24).

The FS method Φ described in Theorem 11 is essentially a backward-elimination algorithm. With slight modifications, the above shows that many popular FS methods that implement variants of backward-search, for example Recursive Feature Elimination (Guyon et al., 2002), are in fact consistent. This provides important evidence of the soundness of these algorithms.

In contrast, forward-search algorithms are not consistent even for strictly positive f. Starting a with feature set S, forward-search would find the feature set $S' = S \cup \{X_i\}$ (that is, add feature X_i) iff $\hat{c}(D_{S'}^l) < \hat{c}(D_S^l)$. But it may happen that $R(g_{S'}^*) \not< R(g_S^*)$ even though S' is contained in S^*. Therefore, forward-search may miss features in S^*. The "noisy XOR problem" (Guyon and Elisseeff, 2003, pp. 1116) is an example of a strictly positive distribution with this property.

A simple example illustrating Theorem 11 is shown in Figure 4. We implemented the feature selection method Φ defined in the theorem, and again used the data density f from Equation (4). Also here, we employed a linear SVM as inducer. We used the leave-one-out error estimate (Devroye et al., 1996) as $\hat{R}$. As sample size increases, we find that the fraction of strongly relevant features selected approaches 1, confirming that $\Phi(D^l) \xrightarrow{P} S^*$. Again, this emphasizes that asymptotic results can serve as good approximations for reasonably large sample sizes.

The algorithm Φ is primarily intended as a constructive proof of the fact that polynomial and consistent algorithms exist; we do not contend that it is optimal in practical situations. Nevertheless,

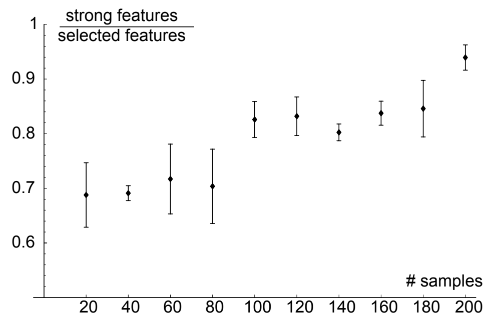


Figure 4: A feature selection example on a 10-dimensional density with 5 strongly and 5 weakly relevant features (Equation 4). Averaged results of 50 runs are plotted for samples sizes $20,\ldots,200$. Error bars denote standard deviations.

Data set	$l \times n$	No FS	$\Phi_{\varepsilon=0}$	RELIEF	FCBF
Breast cancer	569×30	8	9(7)	$\underline{69}$(8)	$\underline{62}$(2)
Ionosphere	351×34	11	9(14)	16(26)	14(5)
Liver disorder	345×6	36	39(5)	$-$(0)	43(2)
E.Coli	336×7	36	$\underline{20}$(5)	43(1)	$\underline{57}$(1)
P.I. Diabetes	768×8	33	36(7)	35(1)	35(1)
Spambase	4601×57	12	17(39)	$\underline{25}$(26)	$\underline{40}$(4)

Table 1: Feature selection on UCI data sets. Test error rates are given in %, number of features selected in parentheses. Significant differences from the classifier without feature selection ("No FS") are underscored (McNemar's test, $p = 0.05$). l denotes number of samples, n number of features.

we conducted some experiments using Φ on a set of well-known data sets from the UCI machine learning repository (Newman et al., 1998) to demonstrate empirically that weakly relevant features do not contribute to classifier accuracy. We used a 5-NN classifier together with a 10-fold cross-validation error estimate for the criterion function $\hat{c}$. For each case we estimated the final accuracy by holding out a test set of 100 examples. Statistical significance was evaluated using McNemar's test (Dietterich, 1998). We set $\varepsilon = 0$ in this test, as we were not particularly concerned about false positives. For comparison we also tried the RELIEF algorithm (Kira. and Rendell, 1992) and the FBCF algorithm by Yu and Liu (2004), both of which are based on the conjecture that weakly relevant features may be needed. We found that Φ never increased test error significantly compared to the full data set, and significantly improved the accuracy in one case (Table 1). The FCBF and Relief algorithms significantly increased the test error in five cases. Overall, these methods selected very few features (in one case, RELIEF selected no features at all) using the default thresholds recommended by the original papers (for FCBF, $\gamma = n/\log n$ and for RELIEF, $\theta = 0$, in the notation of each respective paper; these correspond to the ε parameter of the Φ algorithm).

In the case of the P.I. Diabetes set, Φ seems to select redundant features, which at first might seem to contradict our theory. This may happen for two reasons. First, at $\varepsilon = 0$, the Φ algorithm is inclined to to include false positives (redundant features) rather than risk any false negatives. Second, it is possible that some of these features are truly in S^*, even though the reduction in classification error is too small to be visible on a small test set.

Theorem 10 also has implications for algorithmic complexity in the case of soft classification. To find the Markov boundary, one need now only test each X_i for strong relevance, that is, for the conditional independence $Y \perp X_i | X_{R_i}$. This procedure is clearly consistent and can be implemented in polynomial time. It is not very practical though, since these tests have very limited statistical power for large n due to the large conditioning sets R_i. However, realistic solutions have recently been devised for more narrow distribution classes such as PC, yielding polynomial and consistent algorithms (Peña et al., 2005; Tsamardinos and Aliferis, 2003).

4. The All-Relevant Problem

Recently, feature selection has received much attention in the field of bioinformatics, in particular in gene expression data analysis. Although classification accuracy is an important objective also in this field, many researchers are more interested in the ”biological significance” of features (genes) that depend on the target variable Y (Slonim, 2002). As a rule, biological significance means that a gene is causally involved in the biological process of interest. It is imperative to understand that this biological significance is very different from that in Definition 5. Clearly, genes may be useful for predicting Y without being causally related, and may therefore be irrelevant to the biologist.

Typically, feature selection is performed to optimize classification performance (that is, one attempts to solve MINIMAL-OPTIMAL), and the features chosen are then examined for biological significance (Golub et al., 1999; Guyon et al., 2002). Unfortunately, this strategy ignores the distinction between biological significance and prediction. The features in S^* are typically those with good signal-to-noise ratio (that is, those very predictive of the class), but these need not be more biologically significant than other features dependent on Y. For example, a biologically very important class of genes called *transcription factors* are often present in very small amounts and are therefore difficult to detect with microarrays, leading to poor signal-to-noise ratios (Holland, 2002). Yet, these genes are often implicated in for example cancer development (Darnell, 2002).

Therefore, it is desirable to identify *all* genes relevant to the target variable, rather than the set S^*, which may be more determined by technical factors than by biological significance. Hence, we suggest that the following feature set should be found and examined.

Definition 12 *For a given data distribution, the* ALL-RELEVANT *feature set S^A is the set of features relevant to Y in the sense of Definition 5.*

To the best of our knowledge, this problem has not been studied. We will demonstrate that because S^A includes weakly relevant features (Figure 3), the ALL-RELEVANT problem is much harder than MINIMAL-OPTIMAL. In fact, the problem of determining whether a single feature X_i is weakly relevant requires exhaustive search over all 2^n subsets of X, even if we restrict ourselves to strictly positive distributions.

Theorem 13 *For a given feature X_i and for every $S \subseteq R_i$, there exists a strictly positive $f(x,y)$ satisfying*

$$Y \not\perp X_i | S \quad \wedge \quad \forall S' \neq S : Y \perp X_i | S'. \tag{7}$$

Proof Without loss of generalization we may take $i = n$ and $S = X_1, \dots X_k$, $k = |S|$. Let $S \cup \{X_{k+1}\}$ be distributed as a $k+1$-dimensional Gaussian mixture

$$\begin{aligned} f(s, x_{k+1} | y) &= \frac{1}{|M_y|} \sum_{\mu \in M_y} N(s, x_{k+1} | \mu, \Sigma), \\ M_y &= \{\mu \in \{1,0\}^{k+1} : \mu_1 \oplus \cdots \oplus \mu_{k+1} = (y+1)/2\}, \end{aligned}$$

where $\oplus$ is the XOR operator (M_y is well-defined since $\oplus$ is associative and commutative). This distribution is a multivariate generalization of the ”noisy XOR problem” (Guyon and Elisseeff, 2003). It is obtained by placing Gaussian densities centered at the corners of a $k+1$-dimensional hypercube given by the sets M_y, for $y = \pm 1$. It is easy to see that this gives $Y \not\perp X_{k+1} | S$ and $Y \perp X_{k+1} | S'$ if $S' \subset S$. Next, let $X_{i+1} = X_i + \varepsilon$ for $k < i < n$, where ε is some strictly positive noise distribution. Then it holds that $Y \not\perp X_i | S$ for $k < i < n$, and in particular $Y \not\perp X_n | S$. But it is also clear that $Y \perp X_n | S'$ for $S' \supset S$, since every such S' contain a better predictor $X_i, k < i < n$ of Y. Taken together, this is equivalent to (7), and f is strictly positive. ■

This theorem asserts that the conditioning set that satisfies the relation $Y \not\perp X_i | S$ may be completely arbitrary. Therefore, no search method other than exhaustively examining all sets S can possibly determine whether X_i is weakly relevant. Since ALL-RELEVANT requires that we determine this for every X_i, the following corollary is immediate.

Corollary 14 *The all-relevant problem requires exhaustive subset search.*

Exhaustive subset search is widely regarded as an intractable problem, and no polynomial algorithm is known to exist. This fact is illustrative in comparison with Theorem 8; MINIMAL-OPTIMAL is tractable for strictly positive distributions precisely because S^* does *not* include weakly relevant features.

Since the restriction to strictly positive distributions is not sufficient to render ALL-RELEVANT tractable, we must look for additional constraints. In the following sections we propose two different polynomial-time algorithms for finding S^A, and prove their consistency.

Algorithm 1: Recursive independence test (RIT)

```
Input: target node $Y$, features $X$
Let $S = \emptyset$;
foreach $X_i \in X$ do
    if $X_i \not\perp Y | \emptyset$ then
        $S = S \cup \{X_i\}$;
    end
end
foreach $X_i \in S$ do
    $S = S \cup \mathrm{RIT}(X_i, X \setminus S)$;
end
return $S$
```

4.1 Recursive Independence Test

A simple, intuitive method for solving ALL-RELEVANT is to test features pairwise for *marginal* dependencies: first test each feature against Y, then test each feature against every variable found to be dependent on Y, and so on, until no more dependencies are found (Algorithm 1). We refer to this algorithm as Recursive Independence Testing (RIT). We now prove that the RIT algorithm is consistent (converges to S^A) for PCWT distributions, provided that the test used is consistent.

Theorem 15 *For any PCWT distribution, let R denote the set of variables $X_k \in X$ for which there exists a sequence $Z_1^m = \{Z_1, \dots, Z_m\}$ between $Z_1 = Y$ and $Z_m = X_k$ such that $Z_i \not\perp Z_{i+1} | \emptyset$, $i = 1, \dots, m-1$. Then $R = S^A$.*

Proof Let $I = X \setminus R$ and fix any $X_k \in I$. Since $Y \perp X_k | \emptyset$ and $X_i \perp X_k | \emptyset$ for any $X_i \in R$, we have $\{Y\} \cup R \perp I | \emptyset$ by the composition property. Then $Y \perp X_k | S$ for any $S \subset X \setminus \{X_k, Y\}$ by the weak union and decomposition properties, so X_k is irrelevant; hence, $S_A \subseteq R$.

For the converse, fix any $X_k \in R$ and let $Z_1^m = \{Z_1, \dots, Z_m\}$ be a shortest sequence between $Z_1 = Y$ and $Z_m = X_k$ such that $Z_i \not\perp Z_{i+1} | \emptyset$ for $i = 1, \dots, m-1$. Then we must have $Z_i \perp Z_j | \emptyset$ for $j > i+1$, or else a shorter sequence would exist. We will prove that $Z_1 \not\perp Z_m | Z_2^{m-1}$ for *any* such shortest sequence, by induction over the sequence length. The case $m = 2$ is trivial. Consider the case $m = p$. Assume as the induction hypothesis that, for any $i, j < p$ and any chain Z_i^{i+j} of length j, it holds that $Z_i \not\perp Z_{i+j} | Z_{i+1}^{i+j-1}$. By the construction of the sequence Z_1^m it also holds that

$$Z_1 \perp Z_i | \emptyset, \quad 3 \le i \le m \quad \underset{\text{(composition)}}{\Longrightarrow} \quad Z_1 \perp Z_3^i | \emptyset \tag{8}$$

$$\underset{\text{(weak union)}}{\Longrightarrow} \quad Z_1 \perp Z_i | Z_3^{i-1}. \tag{9}$$

Now assume to the contrary that $Z_1 \perp Z_p | Z_2^{p-1}$. Together with (9), weak transitivity implies

$$Z_1 \perp Z_2 | Z_3^{p-1} \quad \vee \quad Z_2 \perp Z_p | Z_3^{p-1}.$$

The latter alternative contradicts the induction hypothesis. The former together with (8) implies $Z_1 \perp Z_2^{p-1} | \emptyset$ by contraction, which implies $Z_1 \perp Z_2 | \emptyset$ by decomposition. This is also a contradiction;

Algorithm 2: Recursive Markov Boundary (RMB)

```
Input: target node $Y$, data $X$, visited nodes $V$
Let $S = M(Y|X)$, the Markov blanket of $Y$ in $X$;
foreach $X_i \in S \setminus V$ do
    $S = S \cup \mathrm{RMB}(Y, X \setminus \{X_i\}, V)$;
    $V = V \cup S$
end
return $S$
```

hence $Z_1 \not\perp Z_p | Z_2^{p-1}$, which completes the induction step. Thus X_k is relevant and $R \subseteq S^A$. The theorem follows. ■

Corollary 16 *For any PCWT distribution and any consistent marginal independence test, the RIT algorithm is consistent.*

Proof Since the test is consistent, the RIT algorithm will discover every sequence $Z_1^m = \{Z_1, \dots, Z_m\}$ between $Z_1 = Y$ and $Z_m = X_k$ with probability 1 as $l \to \infty$. Consistency follows from Theorem 15. ■

Since the RIT algorithm makes up to $n = |X|$ tests for each element of R found, in total RIT will evaluate no more than $n|R|$ tests. Thus, for small R the number of tests is approximately linear in n, although the worst-case complexity is quadratic.

There are many possible alternatives as to what independence test to use. A popular choice in Bayesian networks literature is Fisher's Z-test, which tests for linear correlations and is consistent within the family of jointly Gaussian distributions (Kalisch and Bühlmann, 2005). Typically, for discrete Y a different test is needed for testing $Y \perp X_i | \emptyset$ than for testing $X_i \perp X_j$. A reasonable choice is Student's t-test, which is consistent for jointly Gaussian distributions (Casella and Berger, 2002). More general independence tests can be obtained by considering correlations in kernel Hilbert spaces, as described by Gretton et al. (2005).

4.2 Recursive Markov Boundary

In this section we propose a second algorithm for ALL-RELEVANT called Recursive Markov Boundary (RMB), based on a given consistent estimator of Markov boundaries of Y. Briefly, the RMB algorithm first estimates $M(Y|X)$, then estimates $M(Y|X \setminus \{X_i\})$ for each $X_i \in M(Y|X)$, and so on recursively until no more nodes are found (Algorithm 2). For efficiency, we also keep track of previously visited nodes V to avoid visiting the same nodes several times. We start the algorithm with $\mathrm{RMB}(Y, X, V = \emptyset)$. A contrived example of the RMB algorithm for a PD distribution is given in Figure 5.

Next, we prove that also the RMB algorithm is consistent for any PCWT distribution, assuming that we are using a consistent estimator of Markov boundaries (see below). The proof makes use of the following concept.

Definition 17 *An independence map (I-map) over a set of features $X = \{X_1, \ldots, X_n\}$ is an undirected graph G over X satisfying*

$$R \perp_G S|T \implies R \perp S|T$$

where R, S, T are disjoint subsets of X and $\perp_G$ denotes vertex separation in the graph G, that is, $R \perp_G S|T$ holds iff every path in G between R and S contains at least one $X_i \in T$. An I-map is minimal if no subgraph G' of G (over the same nodes X) is an I-map.

Note that we only consider the minimal I-map over the features X (not over $X \cup \{Y\}$), since in this case, the minimal I-map is unique for any strictly positive distribution (Pearl, 1988).

Theorem 18 *For any PCWT distribution such that a given estimator $M(Y|S)$ of the Markov boundary of Y with respect to a feature set S is consistent for every $S \subseteq X$, the RMB algorithm is consistent.*

Proof For every $S \subseteq X$, the marginal distribution over S, Y is strictly positive, and therefore every Markov boundary $M(Y|S)$ is unique by corollary 26 (Appendix B). Let G be the minimal I-map over the features X, and let $M_1 = M(X|Y)$. Fix an X_k in S^A. If $X_k \in M_1$, we know that X_k is found by RMB. Otherwise, by Lemma 22, there exists a shortest path Z_1^m in G between some $Z_1 \in M_1$ and $Z_m = X_k$. We prove by induction over m that RMB visits every such path. The case $m = 1$ is trivial. Let the induction hypothesis be that Z_p is visited. For Z_{p+1}, Lemma 22 implies $Y \not\perp Z_{p+1}|X \setminus Z_1^{p+1}$. Since Z_p is visited, RMB will also visit all nodes in $M_{p+1} = M(Y|X \setminus Z_1^p)$. However, M_{p+1} contains Z_{p+1}, because it contains all X_i satisfying $Y \not\perp X_i|X \setminus Z_1^p \setminus \{X_i\}$ by Theorem 10. ■

It is easy to see from algorithm 2 that the RMB algorithm requires computing $|S_A|$ Markov blankets. We might attempt to speed it up by marginalizing out several nodes at once, but in that case we cannot guarantee consistency. A general algorithm for estimating Markov boundaries is given by Peña et al. (2005). This estimator is consistent in the PC class, so it follows that RMB is consistent in PCWT in this case (see Figure 1).

At first sight, RMB may seem more computationally intensive that RIT. However, since the Markov boundary is closely related to S^* (see Section 3.2), we anticipate that RMB may be implemented using existing FS methods. In particular, for distribution classes where the Markov boundary coincides with the MINIMAL-OPTIMAL set S^*, one may compute $M(Y|X)$ using any FS method that consistently estimates S^*. For example, this property holds for the well-known class of jointly Gaussian distributions $f(x|y) = N(x|y\mu, \Sigma)$ with $p(y) = 1/2$. To see this, note that the posterior and Bayes classifier are given by

$$\begin{aligned} p(y|x) &= \left[1 + \exp\{-2y\mu^T\Sigma^{-1}x\}\right]^{-1}, \\ g^*(x) &= \operatorname{sgn}(\mu^T\Sigma^{-1}x). \end{aligned}$$

Clearly, both $g^*(x)$ and $p(y|x)$ are constant with respect to an x_i iff $(\mu^T\Sigma^{-1})_i = 0$. Thus, in this case S^* equals the Markov boundary. SVM-based FS methods are one attractive option, as there exist efficient optimization methods for re-computation of the SVM solution after marginalization (Keerthi, 2002).

4.3 Related Work

We have not been able to find any previous work directly aimed at solving the ALL-RELEVANT problem. It is related to inference of graphical probability models: for the PD distributions class,

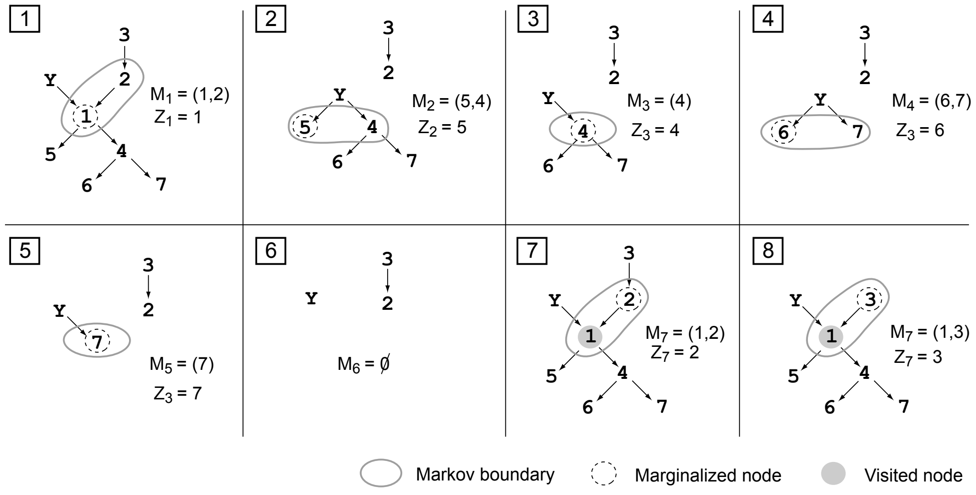


Figure 5: A contrieved example of the RMB algorithm for a PD distribution faithful to the DAG shown in black arrows. Numbers denote the relevant features, Y denotes the target variable. As in Theorem 18, M_i denotes Markov boundaries and Z_i denotes marginalized nodes. Note that the marginal distributions from step 2 onwards may not be PD, so absence of arrows should not be interpreted as independencies.

ALL-RELEVANT can be solved by inferring a Bayesian network and then taking S_A to be the connected component of Y in that network. However, this is less efficient than our approach, since Bayesian network inference is asymptotically NP-hard, even in the very restricted PD class (Chickering et al., 2004). Certainly, such a strategy seems inefficient as it attempts to ”solve a harder problem as an intermediate step” (by inferring a detailed model of the data distribution merely to find the set S^A), thus violating Vapnik's famous principle (Vapnik, 2000, pp. 39).

On the other hand, several methods have been proposed for solving MINIMAL-OPTIMAL that in fact attempt to find all relevant features, since they do not assume $f > 0$ and therefore cannot rule out the weakly relevant ones. These include FOCUS (Almuallim and Dietterich, 1991), which considers the special case of binary $\mathcal{X}$ and noise-free labels; RELIEF (Kira. and Rendell, 1992), a well-known approximate procedure based on nearest-neighbors; Markov blanket filtering (Koller and Sahami, 1996; Yu and Liu, 2004), which considers the special case of marginal dependencies (and is therefore fundamentally different from RMB, despite the similar name). All known methods are either approximate or have exponential time-complexity.

5. Conclusion

In this paper, we have explored an alternative approach to the feature selection (FS) problem: instead of designing suboptimal methods for the intractable full problem, we propose consistent and efficient (polynomial-time) methods for a restricted data distribution class. We find that a very mild restriction to *strictly positive* distributions is sufficient for the MINIMAL-OPTIMAL problem to be tractable (Figure 1). Therefore, we conclude that it is tractable in most practical settings.

We have also identified a different feature selection problem, that of discovering all relevant features (ALL-RELEVANT). This problem is much harder than MINIMAL-OPTIMAL, and has hitherto received little attention in the machine learning field. With the advent of major new applications in the bioinformatics field, where identifying features *per se* is often a more important goal than building accurate predictors, we anticipate that ALL-RELEVANT will become a very important research problem in the future. We have herein provided a first analysis, proved that the problem is intractable even for strictly positive distributions, and proposed two consistent, polynomial-time algorithms for more restricted classes (Figure 1). We hope that these results will inspire further research in this novel and exciting direction.

Acknowledgments

We would like to thank the editor and the anonymous reviewers for helpful comments and insight, as well as for pointing out an initial problem with the proof of Lemma 19 and also suggesting a remedy to that problem. We also thank Timo Koski, Department of Mathematics, Linköping University for proof-reading and Peter Jonsson, Department of Computer and Information Science, Linköping University for valuable discussions. This work was supported by grants from the Swedish Foundation for Strategic Research (SSF), the Ph.D. Programme in Medical Bioinformatics, the Swedish Research Council (VR-621-2005-4202), Clinical Gene Networks AB and Linköping University.

Appendix A. Lemmas

For completeness, we here give a proof of the uniqueness of the Bayes classifier for strictly positive distributions.

Lemma 19 *For any strictly positive distribution $f(x,y)$, the Bayes classifier g^* is unique in the sense that, for every classifier g, it holds that $R(g) = R(g^*)$ iff the Lebesgue measure of $\{x : g(x) \neq g^*(x)\}$ is zero.*

Proof By Theorem 2.2 of Devroye et al. (1996), the risk of any classifier g can be written as

$$R(g) = R(g^*) + \int_X \left| \max_y p(y|x) - \frac{1}{2} \right| 1_{\{g(x) \neq g^*(x)\}} f(x) dx.$$

By property (ii) of Definition 1, we have $\max_y p(y|x) \neq 1/2$ almost everywhere. Thus, the integral is zero iff the Lebesgue measure of $\{x : g(x) \neq g^*(x)\}$ is zero. ■

Lemma 20 *For any conditional distribution $p(y|x)$, it holds that*

$$P(p(Y|X_i,R_i) = p(Y|X_i',R_i)) = 1 \iff P(p(Y|X_i,R_i) = p(Y|R_i)) = 1$$

provided that X_i, X_i' are independent and identically distributed.

Proof Assume that the left-hand side holds. Then we must have

$$P(p(Y|X_i,R_i)=p_0)=1$$

for some p_0 constant with respect to X_i. But

$$p(y|r_i)=\frac{f(r_i,y)}{f(r_i)}=\frac{\int_{X_i}p(y|x)f(x)}{\int_{X_i}f(x)}=\frac{p_0\int_{X_i}f(x)}{\int_{X_i}f(x)}=p_0$$

with probability 1, which implies the right-hand side. The converse is trivial. ■

The following lemmas are needed for the correctness proof for the RMB algorithm.

Lemma 21 *Let $f(x)$ be any PCWT distribution, so that a unique undirected minimal I-map G of f exists. Then, for any shortest path $Z_1^m=\{Z_1,\dots Z_m\}$ between Z_1 and Z_m in G, it holds that $Z_1 \not\perp Z_m|X\setminus Z_1^m$.*

Proof The proof is by induction. For $m=2$, the lemma follows immediately from the definition of the minimal I-map (Pearl, 1988). Also, it holds that

$$Z_i \not\perp Z_{i+1}|X\setminus\{Z_i,Z_{i+1}\} \tag{10}$$

$$Z_i \perp Z_j|X\setminus\{Z_i,Z_j\},\quad j>i+1. \tag{11}$$

Take any distinct Z_i^{i+1},Z_k and assume that $Z_i\perp Z_{i+1}|X\setminus\{Z_i,Z_{i+1},Z_k\}$. Then $Z_i\perp\{Z_{i+1},Z_k\}|X\setminus\{Z_i,Z_{i+1},Z_k\}$ by contraction with (11), and therefore $Z_i\perp Z_{i+1}|X\setminus\{Z_i,Z_{i+1}\}$ by weak union. This contradicts (10), so we conclude that

$$Z_i \not\perp Z_{i+1}|X\setminus\{Z_i,Z_{i+1},Z_k\}. \tag{12}$$

Next, take any sequence Z_i^{i+2}. Applying (12), we obtain $Z_i\not\perp Z_{i+1}|X\setminus Z_i^{i+2}$ and $Z_{i+1}\not\perp Z_{i+2}|X\setminus Z_i^{i+2}$. Using weak transitivity implies either $Z_i\not\perp Z_{i+2}|X\setminus Z_i^{i+2}$ or $Z_i\not\perp Z_{i+2}|X\setminus\{Z_i,Z_{i+2}\}$. The latter alternative contradicts (11), so we conclude

$$Z_i \not\perp Z_{i+2}|X\setminus Z_i^{i+2}. \tag{13}$$

Finally, take any Z_i,Z_j,Z_k such that neither Z_i,Z_j nor Z_j,Z_k are consecutive in the path Z_1^m. Using (11) with intersection (Theorem 25) and decomposition (Theorem 23), we find

$$\left.\begin{array}{l} Z_i\perp Z_j|X\setminus\{Z_i,Z_j\} \\ Z_j\perp Z_k|X\setminus\{Z_j,Z_k\} \end{array}\right\} \implies Z_i\perp Z_j|X\setminus\{Z_i,Z_j,Z_k\}. \tag{14}$$

Equations (12),(13) and (14) show that the properties (10) and (11) hold also for the shortened path $Z_1',\dots,Z_{m-1}'$ given by $Z_1'=Z_1$ and $Z_i'=Z_{i+1}, 2\le i<m$ (removing Z_2). The lemma follows from (10) by induction. ■

Lemma 22 *For any PCWT distribution $f(x,y)$, a feature X_k is relevant iff there exists a path $Z_1^m=\{Z_1,\dots Z_m\}$ in the minimal I-map of $f(x)$ between some $Z_1\in M=M(Y|X)$ and $Z_m=X_k$. In particular, for such a path it holds that $Y\not\perp Z_m|X\setminus Z_1^m$.*

Proof If $Z_m \in M$ (that is, $m = 1$), the lemma is trivial. Consider any $Z_m \notin M$. First, assume that there exists no path Z_1^m. Then $Z_m \perp M|S$ for any $S \subseteq X \setminus M \setminus \{Z_m\}$ by Lemma 21. Fix such an S. Since $Z_m \perp Y|M \cup S$, contraction and weak union gives $Z_m \perp M|\{Y\} \cup S$. Again using $Z_m \perp M|S$, weak transitivity gives

$$Z_m \perp Y|S \quad \vee \quad Y \perp M|S.$$

The latter alternative is clearly false; we conclude $Z_m \perp Y|S$. Next, fix any $S' \subseteq M$. By decomposition, $Z_m \perp M|S \implies Z_m \perp S'|S$. Combining with the above result, by the composition property

$$\left.\begin{array}{l} Z_m \perp S'|S \\ Z_m \perp Y|S \end{array}\right\} \implies Z_m \perp \{Y\} \cup S'|S.$$

Finally, weak union gives $Z_m \perp Y|S \cup S'$. Since $S \cup S'$ is any subset of $X \setminus \{Z_m\}$, we conclude that Z_m is irrelevant.

For the converse, assume that there exists a path Z_1^m. By Lemma 21, we have $Z_1 \not\perp Z_m|X \setminus Z_1^m$. Also, since $Z_1 \in M$ and $Z_2^m \cap M = \emptyset$, it holds that $Y \not\perp Z_1|S$ for any S that contains $M \setminus \{Z_1\}$. In particular, take $S = X \setminus Z_1^m$. Weak transitivity then gives

$$\left.\begin{array}{l} Z_1 \not\perp Z_m|X \setminus Z_1^m \\ Y \not\perp Z_1|X \setminus Z_1^m \end{array}\right\} \implies Z_m \not\perp Y|X \setminus Z_1^m \quad \vee \quad Z_m \not\perp Y|X \setminus Z_2^m.$$

But the latter alternative is false, since $X \setminus Z_2^m$ contains M by assumption. We conclude that $Z_m \not\perp Y|X \setminus Z_1^m$ and that Z_m is relevant. ■

Appendix B. Distribution Classes and Properties

The following two theorems are given by Pearl (1988) and concern any probability distribution.

Theorem 23 *Let R, S, T, U denote any disjoint subsets of variables. Any probability distribution satisfies the following properties:*

$$\begin{array}{l}
\textit{Symmetry:} \quad S \perp T|R \implies T \perp S|R \\
\textit{Decomposition:} \quad S \perp T \cup U|R \implies S \perp T|R \\
\textit{Weak union:} \quad S \perp T \cup U|R \implies S \perp T|R \cup U \\
\textit{Contraction:} \quad S \perp T|R \cup U \wedge S \perp U|R \implies S \perp T \cup U|R
\end{array}$$

Theorem 24 *Let R, S, T, U denote any disjoint subsets of variables and let γ denote a single variable. Any DAG-faithful probability distribution satisfies the following properties:*

$$\begin{array}{l}
\textit{Composition:} \quad S \perp T|R \wedge S \perp U|R \implies S \perp T \cup U|R \\
\textit{Weak transitivity:} \quad S \perp T|R \wedge S \perp T|R \cup \gamma \implies S \perp \gamma|R \vee \gamma \perp T|R
\end{array}$$

The next theorem is a slight modification of that found in Pearl (1988), adapted to our classification setting and our Definition 1.

Theorem 25 *Let R, S, T be disjoint subsets of X and let Y be the target variable. Any strictly positive distribution $f(x,y)$ satisfies the intersection property*

$$Y \perp R|(S \cup T) \wedge Y \perp T|(S \cup R) \Rightarrow Y \perp (R \cup T)|S.$$

Proof Using $Y \perp R|(S \cup T)$ we find

$$\begin{aligned} f(r,s,t,y) &= p(y|r,s,t)f(r,s,t) \\ &= p(y|s,t)f(r,s,t). \end{aligned}$$

Similarly, from $Y \perp T|(S \cup R)$ we find that $f(r,s,t,y) = p(y|s,r)f(r,s,t)$. Because $f(r,s,t) > 0$, it follows that $p(y|s,r) = p(y|s,t)$. Therefore both of these probabilities must be constant w.r.t. R and T, that is,

$$p(y|s,t) = p(y|s,r) = p(y|s).$$

Hence, $Y \perp R|S$ and $Y \perp T|S$ holds. The intersection property follows by contraction. ∎

Corollary 26 *For any strictly positive distribution $f(x,y)$, the Markov boundary $M(Y|X)$ is unique.*

Proof Let S be the set of all Markov blankets of Y, $S = \{T \subseteq X : Y \perp X \setminus T|T\}$. Let T_1, T_2 be any two Markov blankets in S. By Theorem 25 the intersection property holds, so with $T' = T_1 \cap T_2$ we obtain

$$\left\{ \begin{array}{l} Y \perp X \setminus T_1|T' \cup (T_1 \setminus T') \\ Y \perp X \setminus T_2|T' \cup (T_2 \setminus T') \end{array} \right. \Longrightarrow Y \perp X \setminus T'|T'$$

Hence T' is a Markov blanket of Y. Continuing in this fashion for all members of S, we obtain the unique $M(Y|X) = T_1 \cap T_2 \cap \cdots \cap T_{|S|}$. ∎

References

Hussein Almuallim and Thomas G. Dietterich. Learning with many irrelevant features. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, volume 2, pages 547–552, 1991. AAAI Press.

David A. Bell and Hui Wang. A formalism for relevance and its application in feature subset selection. *Machine Learning*, 41:175–195, 2000.

Avril L. Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97:245–271, 1997.

George Casella and Roger L. Berger. *Statistical Inference*. Duxbury, 2nd edition, 2002.

David Chickering and Christopher Meek. Finding optimal Bayesian networks. In *Proceedings of the 18th Annual Conference on Uncertainty in Artificial Intelligence*, pages 94–102, San Francisco, CA, 2002. Morgan Kaufmann Publishers.

David Maxwell Chickering, David Heckerman, and Christopher Meek. Large-sample learning of Bayesian networks is NP-hard. *Journal of Machine Learning Research*, 5:1287–1330, 2004.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

Thomas M. Cover and Jan M. van Campenhout. On the possible orderings of the measurement selection problem. *IEEE Transactions on Systems, Man, and Cybernetics*, 7(9):657–661, 1977.

James E. Darnell. Transcription factors as targets for cancer therapy. *Nature Reviews Cancer*, 2: 740–749, 2002.

Luc Devroye, Lázló Györfi, and Gábor Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, New York, 1996.

Thomas G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10:1895–1923, 1998.

Nir Friedman. Inferring Cellular Networks Using Probabilistic Graphical Models. *Science*, 303: 799–805, 2004.

Todd R. Golub, Donna K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, and Eric S. Lander. Molecular classifiation of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.

Arthur Gretton, Ralf Herbrich, Alexander Smola, Olivier Bousquet, and Bernhard Schölkopf. Kernel methods for measuring independence. *Journal of Machine Learning Research*, 6:2075–2129, 2005.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

Douglas P. Hardin, Constantin Aliferis, and Ioannis Tsamardinos. A theoretical characterization of SVM-based feature selection. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

Michael J. Holland. Transcript abundance in yeast varies over six orders of magnitude. *Journal of Biological Chemistry*, 277:14363–66, 2002.

Anil K. Jain and William G. Waller. On the optimal number of features in the classification of multivariate gaussian data. *Pattern Recognition*, 10:365–374, 1978.

George H. John, Ron Kohavi, and Karl Pfleger. Irrelevant features and the subset selection problem. In *Proceedings of the 11th International Conference on Machine Learning*, pages 121–129, 1994.

Markus Kalisch and Peter Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. Technical report, Seminar für Statistik, ETH Zürich, Switzerland, 2005.

S. Sathiya Keerthi. Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms. *IEEE Transactions on Neural Networks*, 13(5):1225–1229, 2002.

Kenji Kira. and Larry A. Rendell. The feature selection problem: Traditional methods and a new algorithm. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 129–134, 1992.

Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97: 273–324, 1997.

Daphne Koller and Mehran Sahami. Towards optimal feature selection. In *Proceedings of the 13th International Conference of Machine Learning*, pages 248–292, 1996.

David J. Newman, S. Hettich, C.L. Blake, and C.J. Merz. UCI repository of machine learning databases, 1998. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Judea Pearl. *Probabilistic reasoning in intelligent systems*. Morgan Kauffman Publishers, Inc., San Fransisco, California, 1988.

José M. Peña, Johan Björkegren, and Jesper Tegnér. Scalable, efficient and correct learning of markov boundaries under the faithfulness assumption. In *Proceedings of the Eighth European Conference on Symbolic and Quantitative Approaches to Reasoning under Uncertainty*, pages 136–147, 2005.

Jose M. Peña, Roland Nilsson, Johan Björkegren, and Jesper Tegnér. Identifying the relevant nodes before learning the structure. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, pages 367–374, 2006.

Donna K. Slonim. From patterns to pathways: Gene expression comes of age. *Nature Genetics*, 32: 502–508, 2002. Supplement.

Ingo Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67–93, 2002.

Milan Studený. *Probabilistic Conditional Independence Structures*. Springer, 1st edition, 2004.

Ioannis Tsamardinos and Constantin Aliferis. Towards principled feature selection: Relevancy, filters and wrappers. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.

Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 2nd edition, 2000.

Lei Yu and Huan Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004.

Journal of Machine Learning Research 8 (2007) 613-636 Submitted 9/06; Revised 1/07; Published 3/07

Estimating High-Dimensional Directed Acyclic Graphs with the PC-Algorithm

Markus Kalisch KALISCH@STAT.MATH.ETHZ.CH
Peter Bühlmann BUHLMANN@STAT.MATH.ETHZ.CH
Seminar für Statistik
ETH Zurich
8092 Zürich, Switzerland

Editor: David Maxwell Chickering

Abstract

We consider the PC-algorithm (Spirtes et al., 2000) for estimating the skeleton and equivalence class of a very high-dimensional directed acyclic graph (DAG) with corresponding Gaussian distribution. The PC-algorithm is computationally feasible and often very fast for sparse problems with many nodes (variables), and it has the attractive property to automatically achieve high computational efficiency as a function of sparseness of the true underlying DAG. We prove uniform consistency of the algorithm for very high-dimensional, sparse DAGs where the number of nodes is allowed to quickly grow with sample size n, as fast as $O(n^a)$ for any $0 < a < \infty$. The sparseness assumption is rather minimal requiring only that the neighborhoods in the DAG are of lower order than sample size n. We also demonstrate the PC-algorithm for simulated data.

Keywords: asymptotic consistency, DAG, graphical model, PC-algorithm, skeleton

1. Introduction

Graphical models are a popular probabilistic tool to analyze and visualize conditional independence relationships between random variables (see Edwards, 2000; Lauritzen, 1996; Neapolitan, 2004). Major building blocks of the models are nodes, which represent random variables and edges, which encode conditional dependence relations of the enclosing vertices. The structure of conditional independence among the random variables can be explored using the Markov properties.

Of particular current interest are directed acyclic graphs (DAGs), containing directed rather than undirected edges, which restrict in a sense the conditional dependence relations. These graphs can be interpreted by applying the directed Markov property (see Lauritzen, 1996). When ignoring the directions of a DAG, we get the skeleton of a DAG. In general, it is different from the conditional independence graph (CIG), see Section 2.1. (Thus, estimation methods for directed graphs cannot be easily borrowed from approaches for undirected CIGs.) As we will see in Section 2.1, the skeleton can be interpreted easily and thus yields interesting insights into the dependence structure of the data.

Estimation of a DAG from data is difficult and computationally non-trivial due to the enormous size of the space of DAGs: the number of possible DAGs is super-exponential in the number of nodes (see Robinson, 1973). Nevertheless, there are quite successful search-and-score methods for problems where the number of nodes is small or moderate. For example, the search space may be restricted to trees as in MWST (Maximum Weight Spanning Trees; see Chow and Liu, 1968; Heckerman et al., 1995), or a greedy search is employed. The greedy DAG search can be improved by exploiting probabilistic equivalence relations, and the search space can be reduced from individual DAGs to equivalence classes, as proposed in GES (Greedy Equivalent Search, see Chickering, 2002a). Although this method seems quite promising when having few or a moderate number of nodes, it is limited by the fact that the space of equivalence classes is conjectured to grow super-exponentially in the nodes as well (Gillispie and Perlman, 2001). Bayesian approaches for DAGs, which are computationally very intensive, include Spiegelhalter et al. (1993) and Heckerman et al. (1995).

An interesting alternative to greedy or structurally restricted approaches is the PC-algorithm (after its authors, Peter and Clark) from Spirtes et al. (2000). It starts from a complete, undirected graph and deletes recursively edges based on conditional independence decisions. This yields an undirected graph which can then be partially directed and further extended to represent the underlying DAG (see later). The PC-algorithm runs in the worst case in exponential time (as a function of the number of nodes), but if the true underlying DAG is sparse, which is often a reasonable assumption, this reduces to a polynomial runtime.

In the past, interesting hybrid methods have been developed. Very recently, Tsamardinos et al. (2006) proposed a computationally very competitive algorithm. We also refer to their paper for a quite exhaustive numerical comparison study among a wide range of algorithms.

We focus in this paper on estimating the equivalence class and the skeleton of DAGs (corresponding to multivariate Gaussian distributions) in the high-dimensional context, that is, the number of nodes p may be much larger than sample size n. We prove that the PC-algorithm consistently estimates the equivalence class and the skeleton of an underlying sparse DAG, as sample size $n \to \infty$, even if $p = p_n = O(n^a)$ $(0 \leq a < \infty)$ is allowed to grow very quickly as a function of n.

Our implementation of the PC-algorithm is surprisingly fast, as illustrated in Section 4.5, and it allows us to estimate a sparse DAG even if p is in the thousands. For the high-dimensional setting with $p \gg n$, sparsity of the underlying DAG is crucial for statistical consistency and computational feasibility. Our analysis seems to be the first establishing a provable correct algorithm (in an asymptotic sense) for high-dimensional DAGs which is computationally feasible.

The question of consistency of a class of methods including the PC algorithm has been treated in Spirtes et al. (2000) and Robins et al. (2003) in the context of causal inference. They show that, assuming only faithfulness (explained in Section 2), uniform consistency cannot be achieved, but pointwise consistency can. In this paper, we extend this in two ways: We provide a set of assumptions which renders the PC-algorithm to be uniformly consistent. More importantly, we show that consistency holds even as the number of nodes and neighbors increases and the size of the smallest non-zero partial correlations decrease as a function of the sample size. Stricter assumptions than the faithfulness condition that render uniform consistency possible have been also proposed in Zhang and Spirtes (2003). A rather general discussion on how many samples are needed to learn the correct structure of a Bayesian Network can be found in Zuk et al. (2006).

The problem of finding the equivalence class of a DAG has a substantial overlap with the problem of feature selection: If the equivalence class is found, the Markov Blanket of any variable (node) can be read of easily. Given a set of nodes V and suppose that M is the Markov Blanket of node X, then X is conditionally independent of $V \backslash M$ given M. Thus, M contains all and only the relevant features for X. In recent years, many other approaches to feature selection have been developed for high dimensions. See for example Goldenberg and Moore (2004) for an approach dealing with very high dimensions or Ng (1998) for a rather general approach dealing with bounds for generalization errors.

2. Finding the Equivalence Class of a DAG

In this section, we first present the main concepts. Then, we give a detailed description of the PC-algorithm.

2.1 Definitions and Preliminaries

A graph $G = (V, E)$ consists of a set of nodes or vertices $V = \{1, \ldots, p\}$ and a set of edges $E \subseteq V \times V$, that is, the edge set is a subset of ordered pairs of distinct nodes. In our setting, the set of nodes corresponds to the components of a random vector $\mathbf{X} \in \mathbb{R}^p$. An edge $(i, j) \in E$ is called directed if $(i, j) \in E$ but $(j, i) \notin E$; we then use the notation $i \to j$. If both $(i, j) \in E$ and $(j, i) \in E$, the edge is called undirected; we then use the notation $i - j$. A directed acyclic graph (DAG) is a graph G where all edges are directed and not containing any cycle.

If there is a directed edge $i \to j$, node i is said to be a parent of node j. The set of parents of node j is denoted by $pa(j)$. The adjacency set of a node j in graph G, denoted by $adj(G, j)$, are all nodes i which are directly connected to j by an edge (directed or undirected). The elements of $adj(G, j)$ are also called neighbors of or adjacent to j.

A probability distribution P on $\mathbb{R}^p$ is said to be faithful with respect to a graph G if conditional independencies of the distribution can be inferred from so-called d-separation in the graph G and vice-versa. More precisely: consider a random vector $\mathbf{X} \sim P$. Faithfulness of P with respect to G means: for any $i, j \in V$ with $i \neq j$ and any set $\mathbf{s} \subseteq V$,

$$\begin{aligned} & \mathbf{X}^{(i)} \text{ and } \mathbf{X}^{(j)} \text{ are conditionally independent given } \{\mathbf{X}^{(r)};\ r \in \mathbf{s}\} \\ \Leftrightarrow\ & \text{node } i \text{ and node } j \text{ are d-separated by the set } \mathbf{s}. \end{aligned}$$

The notion of d-separation can be defined via moral graphs; details are described in Lauritzen (1996, Prop. 3.25). We remark here that faithfulness is ruling out some classes of probability distributions. An example of a non-faithful distribution is given in Spirtes et al. (2000, Chapter 3.5.2). On the other hand, non-faithful distributions of the multivariate normal family (which we will limit ourselves to) form a Lebesgue null-set in the space of distributions associated with a DAG G, see Meek (1995a).

The skeleton of a DAG G is the undirected graph obtained from G by substituting undirected edges for directed edges. A v-structure in a DAG G is an ordered triple of nodes (i, j, k) such that G contains the directed edges $i \to j$ and $k \to j$, and i and k are not adjacent in G.

It is well known that for a probability distribution P which is generated from a DAG G, there is a whole equivalence class of DAGs with corresponding distribution P (see Chickering, 2002a, Section 2.2). Even when having infinitely many observations, we cannot distinguish among the different DAGs of an equivalence class. Using a result from Verma and Pearl (1990), we can characterize equivalent classes more precisely: Two DAGs are equivalent if and only if they have the same skeleton and the same v-structures.

A common tool for visualizing equivalence classes of DAGs are *completed partially directed acyclic graphs (CPDAG)*. A partially directed acyclic graph (PDAG) is a graph where some edges are directed and some are undirected and one cannot trace a cycle by following the direction of directed edges and any direction for undirected edges. Equivalence among PDAGs or of PDAGs and DAGs can be decided as for DAGs by inspecting the skeletons and v-structures. A PDAG is *completed*, if (1) every directed edge exists also in every DAG belonging to the equivalence class of the DAG and (2) for every undirected edge $i - j$ there exists a DAG with $i \to j$ and a DAG with $i \leftarrow j$ in the equivalence class.

CPDAGs encode all independence informations contained in the corresponding equivalence class. It was shown in Chickering (2002b) that two CPDAGs are identical if and only if they represent the same equivalence class, that is, they represent a equivalence class uniquely.

Although the main goal is to identify the CPDAG, the skeleton itself already contains interesting information. In particular, if P is faithful with respect to a DAG G,

$$\begin{aligned} & \text{there is an edge between nodes } i \text{ and } j \text{ in the skeleton of DAG } G \\ \Leftrightarrow\ & \text{for all } \mathbf{s} \subseteq V \setminus \{i,j\},\ \mathbf{X}^{(i)} \text{ and } \mathbf{X}^{(j)} \text{ are conditionally dependent} \\ & \qquad\qquad\qquad\qquad \text{given } \{\mathbf{X}^{(r)};\ r \in \mathbf{s}\}, \end{aligned} \tag{1}$$

(Spirtes et al., 2000, Th. 3.4). This implies that if P is faithful with respect to a DAG G, the skeleton of the DAG G is a subset (or equal) to the conditional independence graph (CIG) corresponding to P. (The reason is that an edge in a CIG requires only conditional dependence given the set $V \setminus \{i,j\}$). More importantly, every edge in the skeleton indicates some strong dependence which cannot be explained away by accounting for other variables. We think, that this property is of value for exploratory analysis.

As we will see later in more detail, estimating the CPDAG consists of two main parts (which will naturally structure our analysis): (1) Estimation of the skeleton and (2) partial orientation of edges. All statistical inference is done in the first part, while the second is just application of deterministic rules on the results of the first part. Therefore, we will put much more emphasis on the analysis of the first part. If the first part is done correctly, the second will never fail. If, however, there occur errors in the first part, the second part will be more sensitive to it, since it depends on the inferential results of part (1) in greater detail. Therefore, when dealing with a high-dimensional setting (large p, small n), the CPDAG is harder to recover than the skeleton. Moreover, the interpretation of the CPDAG depends much more on the global correctness of the graph. The interpretation of the skeleton, on the other hand, depends only on a local region and is thus more reliable.

We conclude that, if the true underlying probability mechanisms are generated from a DAG, finding the CPDAG is the main goal. The skeleton itself oftentimes already provides interesting insights, and in a high-dimensional setting it might be interesting to use the undirected skeleton as an alternative target to the CPDAG when finding a useful approximation of the CPDAG seems hopeless.

As mentioned before, we will in the following describe two main steps. First, we will discuss the part of the PC-algorithm that leads to the skeleton. Afterwards we will complete the algorithm by discussing the extensions for finding the CPDAG. We will use the same format when discussing theoretical properties of the PC-algorithm.

2.2 The PC-algorithm for Finding the Skeleton

A naive strategy for finding the skeleton would be to check conditional independencies given all subsets $\mathbf{s} \subseteq V \setminus \{i,j\}$ (see Formula 1), that is, all partial correlations in the case of multivariate normal distributions as first suggested by Verma and J.Pearl (1991). This would become computationally infeasible and statistically ill-posed for p larger than sample size. A much better approach is used by the PC-algorithm which is able to exploit sparseness of the graph. More precisely, we apply the part of the PC-algorithm that identifies the undirected edges of the DAG.

2.2.1 POPULATION VERSION FOR THE SKELETON

In the population version of the PC-algorithm, we assume that perfect knowledge about all necessary conditional independence relations is available. We refer here to the PC-algorithm what others call the first part of the PC-algorithm; the other part is described in Algorithm 2 in Section 2.3.

Algorithm 1 The PC$_{pop}$-algorithm

```
1: INPUT: Vertex Set $V$, Conditional Independence Information
2: OUTPUT: Estimated skeleton $C$, separation sets $S$ (only needed when directing the skeleton
   afterwards)
3: Form the complete undirected graph $\tilde{C}$ on the vertex set V.
4: $\ell = -1$;   $C = \tilde{C}$
5: repeat
6:    $\ell = \ell + 1$
7:    repeat
8:       Select a (new) ordered pair of nodes $i$,$j$ that are adjacent in $C$ such that $|adj(C,i) \setminus \{j\}| \geq \ell$
9:       repeat
10:         Choose (new) $\mathbf{k} \subseteq adj(C,i) \setminus \{j\}$ with $|\mathbf{k}| = \ell$.
11:         if $i$ and $j$ are conditionally independent given $\mathbf{k}$ then
12:            Delete edge $i,j$
13:            Denote this new graph by $C$
14:            Save $\mathbf{k}$ in $S(i,j)$ and $S(j,i)$
15:         end if
16:      until edge $i,j$ is deleted or all $\mathbf{k} \subseteq adj(C,i) \setminus \{j\}$ with $|\mathbf{k}| = \ell$ have been chosen
17:   until all ordered pairs of adjacent variables $i$ and $j$ such that $|adj(C,i) \setminus \{j\}| \geq \ell$ and $\mathbf{k} \subseteq$
      $adj(C,i) \setminus \{j\}$ with $|\mathbf{k}| = \ell$ have been tested for conditional independence
18: until for each ordered pair of adjacent nodes $i$,$j$: $|adj(C,i) \setminus \{j\}| < \ell$.
```

The (first part of the) PC-algorithm is given in Algorithm 1. The maximal value of ℓ in Algorithm 1 is denoted by

$$m_{reach} = \text{ maximal reached value of } \ell. \tag{2}$$

The value of m_{reach} depends on the underlying distribution.

A proof that this algorithm produces the correct skeleton can be easily deduced from Theorem 5.1 in Spirtes et al. (2000). We summarize the result as follows.

Proposition 1 *Consider a DAG G and assume that the distribution P is faithful to G. Denote the maximal number of neighbors by* $q = \max_{1 \leq j \leq p} |adj(G,j)|$. *Then, the PC$_{pop}$-algorithm constructs the true skeleton of the DAG. Moreover, for the reached level:* $m_{reach} \in \{q-1, q\}$.

A proof is given in Section A.

2.2.2 SAMPLE VERSION FOR THE SKELETON

For finite samples, we need to estimate conditional independencies. We limit ourselves to the Gaussian case, where all nodes correspond to random variables with a multivariate normal distribution. Furthermore, we assume faithful models, which means that the conditional independence relations correspond to d-separations (and so can be read off the graph) and vice versa; see Section 2.1.

In the Gaussian case, conditional independencies can be inferred from partial correlations.

Proposition 2 *Assume that the distribution P of the random vector* $\mathbf{X}$ *is multivariate normal. For* $i \neq j \in \{1,\dots,p\}$, $\mathbf{k} \subseteq \{1,\dots,p\} \setminus \{i,j\}$, *denote by* $\rho_{i,j|\mathbf{k}}$ *the partial correlation between* $\mathbf{X}^{(i)}$ *and* $\mathbf{X}^{(j)}$ *given* $\{\mathbf{X}^{(r)};\ r \in \mathbf{k}\}$. *Then,* $\rho_{i,j|\mathbf{k}} = 0$ *if and only if* $\mathbf{X}^{(i)}$ *and* $\mathbf{X}^{(j)}$ *are conditionally independent given* $\{\mathbf{X}^{(r)};\ r \in \mathbf{k}\}$.

Proof: The claim is an elementary property of the multivariate normal distribution, see Lauritzen (1996, Prop. 5.2.). □

We can thus estimate partial correlations to obtain estimates of conditional independencies. The sample partial correlation $\hat{\rho}_{i,j|\mathbf{k}}$ can be calculated via regression, inversion of parts of the covariance matrix or recursively by using the following identity: for some $h \in \mathbf{k}$,

$$\rho_{i,j|\mathbf{k}} = \frac{\rho_{i,j|\mathbf{k}\setminus h} - \rho_{i,h|\mathbf{k}\setminus h}\rho_{j,h|\mathbf{k}\setminus h}}{\sqrt{(1-\rho^2_{i,h|\mathbf{k}\setminus h})(1-\rho^2_{j,h|\mathbf{k}\setminus h})}}.$$

In the following, we will concentrate on the recursive approach. For testing whether a partial correlation is zero or not, we apply Fisher's z-transform

$$Z(i,j|\mathbf{k}) = \frac{1}{2}\log\left(\frac{1+\hat{\rho}_{i,j|\mathbf{k}}}{1-\hat{\rho}_{i,j|\mathbf{k}}}\right). \tag{3}$$

Classical decision theory yields then the following rule when using the significance level α. Reject the null-hypothesis $H_0(i,j|\mathbf{k}):\ \rho_{i,j|\mathbf{k}} = 0$ against the two-sided alternative $H_A(i,j|\mathbf{k}):\ \rho_{i,j|\mathbf{k}} \neq 0$ if $\sqrt{n-|\mathbf{k}|-3}|Z(i,j|\mathbf{k})| > \Phi^{-1}(1-\alpha/2)$, where $\Phi(\cdot)$ denotes the cdf of $\mathcal{N}(0,1)$.

The sample version of the PC-algorithm is almost identical to the population version in Section 2.2.1.

The PC-algorithm

Run the PC$_{pop}(m)$-algorithm as described in Section 2.2.1 but replace in line 11 of Algorithm 1 the if-statement by
if $\sqrt{n-|\mathbf{k}|-3}|Z(i,j|\mathbf{k})| \leq \Phi^{-1}(1-\alpha/2)$ **then**.

The algorithm yields a data-dependent value $\hat{m}_{reach,n}$ which is the sample version of (2).

The only tuning parameter of the PC-algorithm is α, which is the significance level for testing partial correlations. See Section 4 for further discussion.

As we will see below in Section 3, the algorithm is asymptotically consistent even if p is much larger than n but the DAG is sparse.

2.3 Extending the Skeleton to the Equivalence Class

While finding the skeleton as in Algorithm 1, we recorded the separation sets that made edges drop out in the variable denoted by S. This was not necessary for finding the skeleton itself, but will be essential for extending the skeleton to the equivalence class. In Algorithm 2 we describe the work of Pearl (2000, p.50f) to extend the skeleton to a CPDAG belonging to the equivalence class of the underlying DAG.

Algorithm 2 Extending the skeleton to a CPDAG

```
  INPUT: Skeleton $G_{skel}$, separation sets $S$
  OUTPUT: CPDAG $G$
  for all pairs of nonadjacent variables $i, j$ with common neighbour $k$ do
    if $k \notin S(i, j)$ then
      Replace $i - k - j$ in $G_{skel}$ by $i \rightarrow k \leftarrow j$
    end if
  end for
  In the resulting PDAG, try to orient as many undirected edges as possible by repeated application
  of the following three rules:
  R1 Orient $j - k$ into $j \rightarrow k$ whenever there is an arrow $i \rightarrow j$ such that $i$ and $k$ are nonadjacent.
  R2 Orient $i - j$ into $i \rightarrow j$ whenever there is a chain $i \rightarrow k \rightarrow j$.
  R3 Orient $i - j$ into $i \rightarrow j$ whenever there are two chains $i - k \rightarrow j$ and $i - l \rightarrow j$ such that $k$ and
  $l$ are nonadjacent.
  R4 Orient $i - j$ into $i \rightarrow j$ whenever there are two chains $i - k \rightarrow l$ and $k \rightarrow l \rightarrow j$ such that $k$ and
  $l$ are nonadjacent.
```

The output of Algorithm 2 is a CPDAG, which was first proved by Meek (1995b).

3. Consistency for High-Dimensional Data

As in Section 2, we will first deal with the problem of finding the skeleton. Consecutively, we will extend the result to finding the CPDAG.

3.1 Finding the Skeleton

We will show that the PC-algorithm from Section 2.2.2 is asymptotically consistent for the skeleton of a DAG, even if p is much larger than n but the DAG is sparse. We assume that the data are realizations of i.i.d. random vectors $\mathbf{X}_1, \ldots, \mathbf{X}_n$ with $\mathbf{X}_i \in \mathbb{R}^p$ from a DAG G with corresponding distribution P. To capture high-dimensional behavior, we will let the dimension grow as a function of sample size: thus, $p = p_n$ and also the DAG $G = G_n$ and the distribution $P = P_n$. Our assumptions are as follows.

(A1) The distribution P_n is multivariate Gaussian and faithful to the DAG G_n for all n.

(A2) The dimension $p_n = O(n^a)$ for some $0 \leq a < \infty$.

(A3) The maximal number of neighbors in the DAG G_n is denoted by
$q_n = \max_{1 \leq j \leq p_n} |adj(G, j)|$, with $q_n = O(n^{1-b})$ for some $0 < b \leq 1$.

(A4) The partial correlations between $\mathbf{X}^{(i)}$ and $\mathbf{X}^{(j)}$ given $\{\mathbf{X}^{(r)}; r \in \mathbf{k}\}$ for some set $\mathbf{k} \subseteq \{1, \ldots, p_n\} \setminus \{i, j\}$ are denoted by $\rho_{n;i,j|\mathbf{k}}$. Their absolute values are bounded from below and above:

$$\inf\{|\rho_{i,j|\mathbf{k}}|;\ i, j, \mathbf{k} \text{ with } \rho_{i,j|\mathbf{k}} \neq 0\} \geq c_n,\ c_n^{-1} = O(n^d), \quad \text{for some } 0 < d < b/2,$$

$$\sup_{n;i,j,\mathbf{k}} |\rho_{i,j|\mathbf{k}}| \leq M < 1,$$

where $0 < b \leq 1$ is as in (A3).

Assumption (A1) is an often used assumption in graphical modeling, although it does restrict the class of possible probability distributions (see also third paragraph of Section 2.1); (A2) allows for an arbitrary polynomial growth of dimension as a function of sample size, that is, high-dimensionality; (A3) is a sparseness assumption and (A4) is a regularity condition. Assumptions (A3) and (A4) are rather minimal: note that with $b = 1$ in (A3), for example fixed $q_n = q < \infty$, the partial correlations can decay as $n^{-1/2+\varepsilon}$ for any $0 < \varepsilon \leq 1/2$. If the dimension p is fixed (with fixed DAG G and fixed distribution P), (A2) and (A3) hold and (A1) and the second part of (A4) remain as the only conditions. Recently, for undirected graphs the Lasso has been proposed as a computationally efficient algorithm for estimating high-dimensional conditional independence graphs where the growth in dimensionality is as in (A2) (see Meinshausen and Bühlmann, 2006). However, the Lasso approach can be inconsistent, even with fixed dimension p, as discussed in detail in Zhao and Yu (2006).

Theorem 1 *Assume (A1)-(A4). Denote by* $\hat{G}_{skel,n}(\alpha_n)$ *the estimate from the (first part of the) PC-algorithm in Section 2.2.2 and by* $G_{skel,n}$ *the true skeleton from the DAG* G_n*. Then, there exists* $\alpha_n \to 0$ $(n \to \infty)$*, see below, such that*

$$\begin{aligned} & \mathbf{P}[\hat{G}_{skel,n}(\alpha_n) = G_{skel,n}] \\ = \ & 1 - O(\exp(-Cn^{1-2d})) \to 1 \ (n \to \infty) \textit{ for some } 0 < C < \infty, \end{aligned}$$

where $d > 0$ *is as in (A4).*

A proof is given in the Section A. A choice for the value of the significance level is $\alpha_n = 2(1 - \Phi(n^{1/2} c_n / 2))$ which depends on the unknown lower bound of partial correlations in (A4).

3.2 Extending the Skeleton to the Equivalence Class

As mentioned before, all inference is done while finding the skeleton. If this part is completed perfectly, that is, if there was no error while testing conditional independencies (it is not enough to assume that the skeleton was estimated correctly), the second part will never fail (see Meek, 1995b). Therefore, we easily obtain:

Theorem 2 *Assume (A1)-(A4). Denote by* $\hat{G}_{CPDAG}(\alpha_n)$ *the estimate from the entire PC-algorithm in Section 2.2.2 and 2.3 and by* G_{CPDAG} *the true CPDAG from the DAG G. Then, there exists* $\alpha_n \to 0$ $(n \to \infty)$*, see below, such that*

$$\begin{aligned} & \mathbf{P}[\hat{G}_{CPDAG}(\alpha_n) = G_{CPDAG}] \\ = \ & 1 - O(\exp(-Cn^{1-2d})) \to 1 \ (n \to \infty) \textit{ for some } 0 < C < \infty, \end{aligned}$$

where $d > 0$ *is as in (A4).*

A proof, consisting of one short argument, is given in the Section A. As for Theorem 1, we can choose $\alpha_n = 2(1-\Phi(n^{1/2}c_n/2))$.

By inspecting the proofs of Theorem 1 and Theorem 2, one can derive explicit error bounds for the error probabilities. Roughly speaking, this bounding function is the product of a linearly increasing and an exponentially decreasing term (in n). The bound is loose but for completeness, we present it in the Appendix.

4. Numerical Examples

We analyze the PC-algorithm for finding the skeleton and the CPDAG using various simulated data sets. The numerical results have been obtained using the `R`-package `pcalg`. For an extensive numerical comparison study of different algorithms, we refer to Tsamardinos et al. (2006).

4.1 Simulating Data

In this section, we analyze the PC-algorithm for the skeleton using simulated data. In order to simulate data, we first construct an adjacency matrix A as follows:

1. Fix an ordering of the variables.

2. Fill the adjacency matrix A with zeros.

3. Replace every matrix entry in the lower triangle (below the diagonal) by independent realizations of Bernoulli(s) random variables with success probability s where $0 < s < 1$. We will call s the sparseness of the model.

4. Replace each entry with a 1 in the adjacency matrix by independent realizations of a Uniform($[0.1, 1]$) random variable.

This then yields a matrix A whose entries are zero or in the range $[0.1, 1]$. The corresponding DAG draws a directed edge from node i to node j if $i < j$ and $A_{ji} \neq 0$. The DAGs (and skeletons thereof) that are created in this way have the following property: $\mathbb{E}[N_i] = s(p-1)$, where N_i is the number of neighbors of a node i.

Thus, a low sparseness parameter s implies few neighbors and vice-versa. The matrix A will be used to generate the data as follows. The value of the random variable $X^{(1)}$, corresponding to the first node, is given by

$$\varepsilon^{(1)} \sim N(0,1),$$
$$X^{(1)} = \varepsilon^{(1)},$$

and the values of the next random variables (corresponding to the next nodes) can be computed recursively as

$$\varepsilon^{(i)} \sim N(0,1),$$
$$X^{(i)} = \sum_{k=1}^{i-1} A_{ik} X^{(k)} + \varepsilon^{(i)} \ (i = 2, \dots, p),$$

where all $\varepsilon^{(1)}, \dots, \varepsilon^{(p)}$ are independent.

4.2 Choice of Significance Level

In Section 3 we provided a value of the significance level $\alpha_n = 2(1 - \Phi(n^{1/2} c_n/2))$. Unfortunately, this value is not constructive, since it depends on the unknown lower bound of partial correlations in (A4). To get a feeling for good values of the significance level in the domain of realistic parameter settings, we fitted a wide range of parameter settings and compared the quality of fit for different significance levels.

Assessing the quality of fit is not quite straightforward, since one has to examine simultaneously both the true positive rate (TPR) and false positive rate (FPR) for a meaningful comparison. We follow an approach suggested by Tsamardinos et al. (2006) and use the Structural Hamming Distance (SHD). Roughly speaking, this counts the number of edge insertions, deletions and flips in order to transfer the estimated CPDAG into the correct CPDAG. Thus, a large SHD indicates a poor fit, while a small SHD indicates a good fit.

We fitted 40 replicates to all combinations of

- $\alpha \in \{0.00005, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1\}$,
- $p \in \{7, 15, 40, 70, 100\}$,
- $n \in \{30, 100, 300, 1000, 3000, 10000, 30000\}$,
- $E[N] \in \{2, 5\}$,

(where $E[N]$ is the average neighborhood size) and evaluated the SHD. Each value of α was used 40 times on each of the 70 possible parameter settings, and we then computed the average SHD over the 70 parameter settings.

The result is shown in Figure 1. One can see that the average SHD achieves a minimum in the region around $\alpha = 0.005$ and $\alpha = 0.01$. For higher or lower significance levels, the average SHD increases; the increase for bigger significance levels is much more pronounced. We analyzed the results of the simulation (see Figure 1) using pairwise Wilcoxon-Tests and Bonferroni correction. It turns out that $\alpha = 0.005$ and $\alpha = 0.01$ yield significantly lower average SHD than the other values of α. In contrast, there is no significant difference between $\alpha = 0.005$ and $\alpha = 0.001$ (without Bonferroni correction). Of course, if n was of different order of magnitude, a reasonable α should be a function of n with $\alpha = \alpha_n \to 0$ $(n \to \infty)$.

4.3 Performance for Different Parameter Settings

In this section, we give an overview over the performance in terms of the true positive rate (TPR) and false positive rate (FPR) for the skeleton and the SHD for the CPDAG. In order to keep the overview at a manageable size, we restrict the significance level to $\alpha = 0.01$. This was one of the two settings minimizing the average SHD as described in the previous section. The remaining parameters will be chosen as follows:

- $p \in \{7, 40, 100\}$,
- $n \in \{30, 100, 300, 1000, 3000, 10000, 30000\}$,
- $E[N] \in \{2, 5\}$.

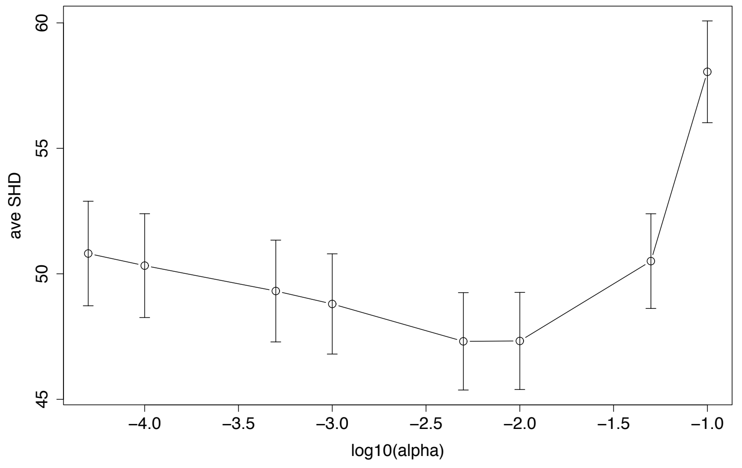


Figure 1: Average Structural Hamming Distance (ave SHD) with 95% confidence intervals. For each value of α, the average SHD was averaged over 70 parameter settings using 40 replicates each. One can see that the average SHD is minimized for significance levels between $\alpha = 0.005$ and $\alpha = 0.01$.

The overview is given in Figure 2. As expected, the fit for a dense graph (triangles; $E[N] = 5$) is worse than the fit for a sparse graph (circles; $E[N] = 2$). While the TPR and the SHD show a clear tendency with increasing sample size, the behavior of FPR is not so clear. The latter seems surprising at first sight but is due to the fact that we used the same $\alpha = 0.01$ for all n.

4.4 Properties in High-Dimensional Setting

In this section, we study the behaviour of the error rates in a high-dimensional setting. The number of variables increases exponentially, the number of samples increases linearly and the expected neighborhood size increases sub-linearly. By inspecting the theory, we would expect the error rates to stay constant or even decrease. Table 1 shows the parameter setting of a small numerical study addressing this question. Note that p increases exponentially, n increases linearly and the expected neighborhood size $E[N] = 0.2\sqrt{n}$ increases sub-linearly. We used $\alpha = 0.05$ and the results are based on 20 simulation runs.

Figure 3 shows boxplots of the TPR and the FPR over 20 replicates of this study. One can easily see that the TPR increases and the FPR decreases with sample size, although the number of $p = p_n$ grows fast and $E[N]$ grows slowly with n. This confirms our theory very clearly.

We should note, that while the number of neighbors to a given variable may be growing almost as fast as n, so that the number of neighbors is increasing with sample size, the percentage of true among all possible edges is going down with n. So in one sense, the sparsity in terms of percentage of true edges of the DAGs is decreasing, and in another sense the sparsity in terms of the neighborhood size is increasing with n.

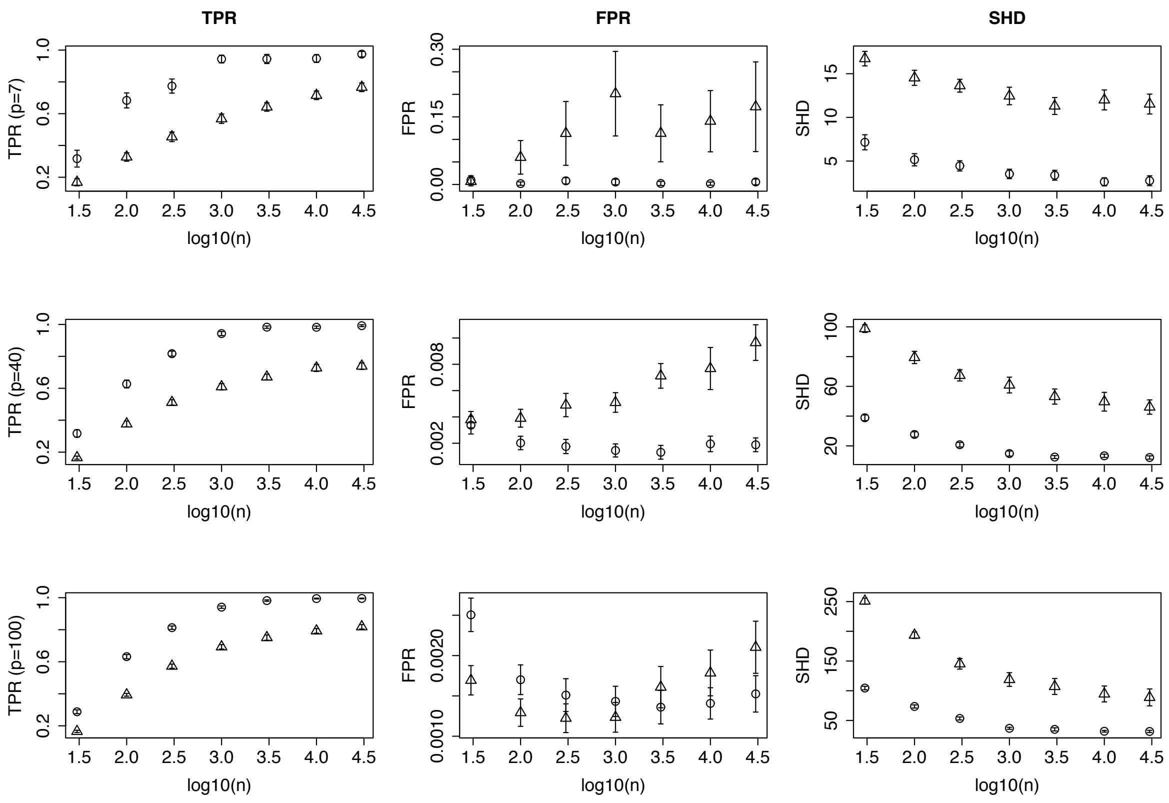


Figure 2: Performance of the PC-algorithm for different parameter settings, showing the mean of TPR, FPR and SHD together with 95% confidence intervals. The triangles represent parameter settings where $E[N] = 5$, while the circles represent parameter settings where $E[N] = 2$.

p	n	$E[N]$	TPR	FPR
9	50	1.4	0.61 (0.03)	0.023 (0.005)
27	100	2.0	0.70 (0.02)	0.011 (0.001)
81	150	2.4	0.753 (0.007)	0.0065 (0.0003)
243	200	2.8	0.774 (0.004)	0.0040 (0.0001)
729	250	3.2	0.794 (0.004)	0.0022 (0.00004)
2187	300	3.5	0.805 (0.002)	0.0012 (0.00002)

Table 1: The number of variables p increases exponentially, the sample size n increases linearly and the expected neighborhood size $E[N]$ increases sub-linearly. As supported by theory, the TPR increases and the FPR decreases in this setting. The results are based on using $\alpha = 0.05$, 20 simulation runs, and standard deviations are given in brackets.

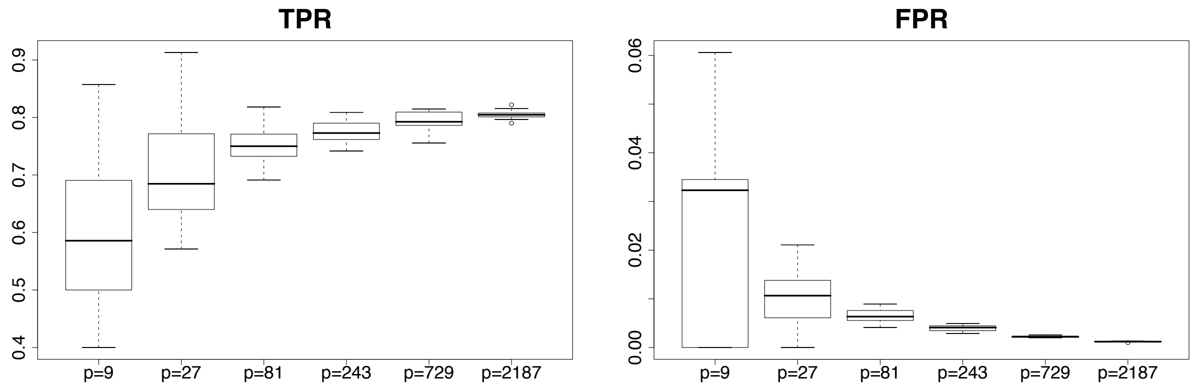


Figure 3: While the number of variables p increases exponentially, the sample size n increases linearly and the expected neighborhood size $E[N]$ increases sub-linearly, the TPR increases and the FPR decreases. See Table 1 for a more detailed specification of the parameters.

4.5 Computational Complexity

Our theoretical framework in Section 3 allows for large values of p. The computational complexity of the PC-algorithm is difficult to evaluate exactly, but the worst case is bounded by

$$O(p^{\hat{m}_{reach}}) \text{ which is with high probability bounded by } O(p^q) \tag{4}$$

as a function of dimensionality p; here, q is the maximal size of the neighborhoods as described in assumption (A3) in Section 3. We note that the bound may be very loose for many distributions. Thus, for the worst case where the complexity bound is achieved, the algorithm is computationally feasible if q is small, say $q \leq 3$, even if p is large. For non-worst cases, however, we can still do the computations for much larger values of q and fairly dense graphs, for example some nodes j having neighborhoods of size up to $|adj(G, j)| = 30$.

We provide a small example of the processor time for estimating a CPDAG by using the PC-algorithm. The runtime analysis was done on an AMD Athlon 64 X2 Dual Core Processor 5000+ with 2.6 GHz and 4 GB RAM running on Linux and using R 2.4.1. The number of variables varied between $p = 10$ and $p = 1000$ while the number of samples was fixed at $n = 1000$. The sparseness was either $E[N] = 2$ or $E[N] = 8$. For each parameter setting, 10 replicates were used. In each case, the significance level used in the PC-algorithm was $\alpha = 0.01$. The average processor time together with its standard deviation for estimating both the skeleton and the CPDAG is given in Table 2. Graphs of $p = 1000$ nodes and 8 neighbors on average could be estimated in about 25 minutes, while networks with up to $p = 100$ nodes could be estimated in about a second. The additional time spent for finding the CPDAG from the skeleton is comparable for both neighborhood sizes and varies between a couple to almost 100 percent of the time needed to estimate the skeleton. The percentage tends to decrease with increasing number of variables.

Figure 4 gives a graphical impression of the results of this example. The sparse graphs (solid line with circles) were estimated faster than the dense graphs. While the line for the dense graph is very straight, the line for the sparse graphs has a positive curvature. Note, that this is a log-log plot; therefore, the slope of the lines indicates the exponent of polynomial growth. In this case, both curves follow very roughly a line with slope two indicating quadratic growth. The positive curvature of the solid line would indicate exponential growth; theory tells us, that this is not possible. One possible explanation for the positive curvature is the fact, that with increasing p, the maximal neighborhood size (which was not controlled in the simulation) is likely to increase. This would gradually increase the exponent in the polynomial growth of the upper bound in Formula (4), thus yielding a positive curvature.

p	$E[N]$	$\hat{G}_{skel}$	$\hat{G}_{CPDAG}$
10	2	0.037 (0.004)	0.072 (0.005)
10	8	0.093 (0.005)	0.124 (0.006)
30	2	0.15 (0.02)	0.23 (0.02)
30	8	0.84 (0.05)	0.93 (0.05)
50	2	0.33 (0.01)	0.48 (0.02)
50	8	2.2 (0.06)	2.4 (0.06)
100	2	1.03 (0.05)	1.49 (0.05)
100	8	8.9 (0.3)	9.4 (0.27)
300	2	8.3 (0.1)	13.8 (0.13)
300	8	89 (3)	95 (3)
1000	2	116 (0.5)	262 (0.8)
1000	8	1300 (60)	1445 (59)

Table 2: The average processor time (Athl. 64, 2.6 GHz, 4 GB) for estimating the skeleton ($\hat{G}_{skel}$) or the CPDAG ($\hat{G}_{CPDAG}$) for different DAGs in seconds, with standard errors in brackets. We used $\alpha = 0.01$ and sample size $n = 1000$.

5. R-Package `pcalg`

The R-package `pcalg` can be used to estimate from data the underlying skeleton or equivalence class of a DAG. To use this package, the statistics software `R` needs to be installed. Both `R` and the R-package `pcalg` are available free of charge at `http://www.r-project.org`. For low-dimensional problems (but not for p in the hundreds or thousands), there are a number of other implementations of the PC-algorithm that are also worth mentioning:

- Hugin at `http://www.hugin.com` .
- Murphy's Bayes Network toolbox at `http://bnt.sourceforge.net` .
- Tetrad IV at `http://www.phil.cmu.edu/projects/tetrad` .

In the following, we show an example of how to generate a random DAG, draw samples and infer from data the skeleton and the equivalence class of the original DAG using the R-package `pcalg`. The line width of the edges in the resulting skeleton and CPDAG can be adjusted to correspond to the reliability of the estimated dependencies. (The line width is proportional to the smallest value of $\sqrt{n-|\mathbf{k}|-3}\, Z(i,j,\mathbf{k})$ causing an edge, see also 3. Therefore, thick lines are reliable).

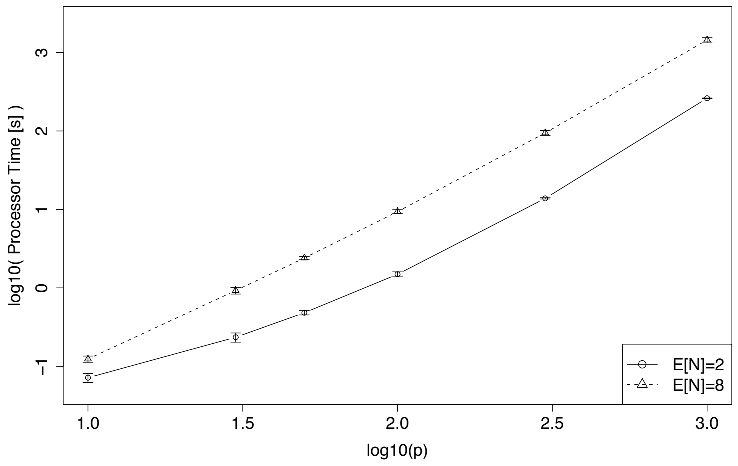


Figure 4: Average processor time over 10 runs together with 95% confidence intervals. Triangles correspond to dense ($E[N] = 8$), circles to sparse ($E[N] = 2$) underlying DAGs. We used $\alpha = 0.01$ and sample size $n = 1000$.

```
library(pcalg)
## define parameters
p <- 10 # number of random variables
n <- 10000 # number of samples
s <- 0.4 # sparsness of the graph
```

For simulating data as described in Section 4.1:

```
## generate random data
set.seed(42)
g <- randomDAG(p,s) # generate a random DAG
d <- rmvDAG(n,g) # generate random samples
```

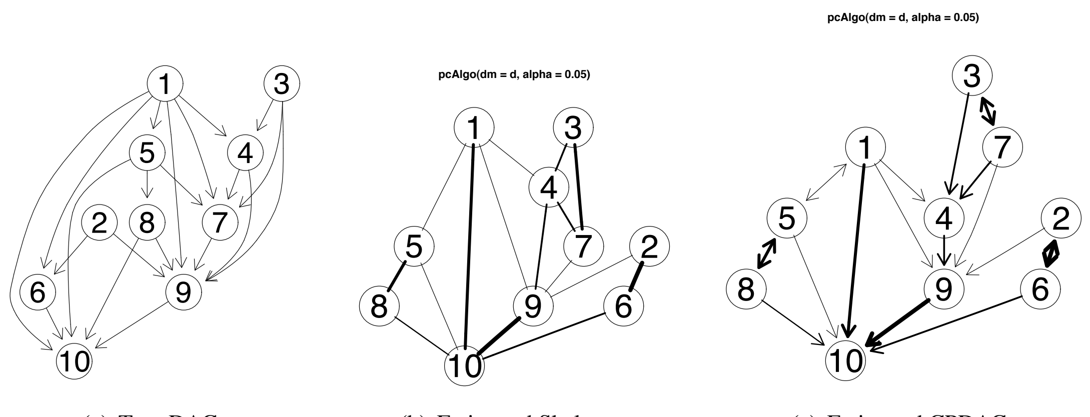
Then we estimate the underlying skeleton by using the function `pcAlgo` and extend the skeleton to the CPDAG by using the function `udag2cpdag`.

```
gSkel <-
  pcAlgo(d,alpha=0.05) # estimate of the skeleton
gCPDAG <-
  udag2cpdag(gSkel)
```

The CPDAG can also be estimated directly using

```
gCPDAG <-
  pcAlgo(d,alpha=0.05, directed=TRUE) # estimate of the CPDAG
```

The results can be easily plotted using the following commands:



(a) True DAG (b) Estimated Skeleton (c) Estimated CPDAG

Figure 5: Plots generated using the R-package pcalg as described in Section 5. (a) The true DAG. (b) The estimated skeleton using the R-function pcAlgo with $\alpha = 0.05$ and $n = 10000$. Line width encodes the reliability (z-values) of the dependence estimates (thick lines are reliable). (c) The estimated CPDAG using the R-function udag2cpdag. Double-headed arrows indicate undirected edges.

```
plot(g)
plot(gSkel,zvalue.lwd=TRUE)
plot(gCPDAG,zvalue.lwd=TRUE)
```

The original DAG is shown in Figure 5(a). The estimated skeleton and the estimated CPDAG are shown in Figure 5(b) and Figure 5(c), respectively. Note the differing line width, which indicates the reliability (z-values as in Formula 3) of the involved statistical tests (thick lines are reliable).

6. Conclusions

We show that the PC-algorithm is asymptotically consistent for the equivalence class of the DAG (represented by the CPDAG) and its skeleton with corresponding very high-dimensional, sparse Gaussian distribution. Moreover, the PC-algorithm is computationally feasible for such high-dimensional, sparse problems. Putting these two facts together, the PC-algorithm is established as a method (so far the only one) which is computationally feasible and provably correct, in the sense of uniform consistency, for high-dimensional DAGs. Sparsity, in terms of the maximal size of the neighborhoods of the true underlying DAG, is crucial for statistical consistency (assumption (A3) and Theorems 1 and 2) and for computational feasibility with at most a polynomial complexity (see Formula 4) as a function of dimensionality.

We emphasize that the skeleton of a DAG oftentimes provides interesting insights, and in a high-dimensional setting it is quite sensible to use the undirected skeleton as a simpler but more realistic target rather than the entire CPDAG. Software for the PC-algorithm is available as explained in Section 5.

Acknowledgments

We would like to thank Martin Mächler for his help with developing the R-package `pcalg`. We also thank three anonymous reviewers and the editor for their constructive comments. Markus Kalisch was supported by the Swiss National Science Foundation (grant no. 200021-105276 and 200020-113270/1).

Appendix A. Proofs

In the following, we give the proofs of all our theorems.

A.1 Proof of Proposition 1

Consider $\mathbf{X}$ with distribution P. Since P is faithful to the DAG G, conditional independence of $\mathbf{X}^{(i)}$ and $\mathbf{X}^{(j)}$ given $\{\mathbf{X}^{(r)};\ r \in \mathbf{k}\}$ ($\mathbf{k} \subseteq V \setminus \{i,j\}$) is equivalent to d-separation of nodes i and j given the set $\mathbf{k}$ (see Spirtes et al., 2000, Th. 3.3). Thus, the population PC$_{pop}$-algorithm as formulated in Section 2.2.1 coincides with the one from Spirtes et al. (2000) which is using the concept of d-separation, and the first claim about correctness of the skeleton follows from Spirtes et al. (2000, Th. 5.1., Ch. 13).

The second claim about the value of m_{reach} can be proved as follows. First, due to the definition of the PC$_{pop}$-algorithm and the fact that it constructs the correct skeleton, $m_{reach} \leq q$. We now argue that $m_{reach} \geq q-1$. Suppose the contrary. Then, $m_{reach} \leq q-2$: we could then continue with a further iteration in the algorithm since $m_{reach}+1 \leq q-1$ and there is at least one node j with neighborhood-size $|adj(G,j)| = q$: that is, the reached stopping level would be at least $q-1$ which is a contradiction to $m_{reach} \leq q-2$. □

A.2 Analysis of the PC-Algorithm

When analyzing the consistency of the PC-algorithm, the convergence properties of partial correlations turn out to be crucial. Therefore, we first concentrate on partial correlations and then shift our focus to the analysis of the PC-algorithm.

A.2.1 ANALYSIS OF PARTIAL CORRELATIONS

We first establish uniform consistency of estimated partial correlations. Denote by $\hat{\rho}_{i,j}$ and $\rho_{i,j}$ the sample and population correlation between $\mathbf{X}^{(i)}$ and $\mathbf{X}^{(j)}$. Likewise, $\hat{\rho}_{i,j|\mathbf{k}}$ and $\rho_{i,j|\mathbf{k}}$ denote the sample and population partial correlation between $\mathbf{X}^{(i)}$ and $\mathbf{X}^{(j)}$ given $\{\mathbf{X}^{(r)}; r \in \mathbf{k}\}$, where $\mathbf{k} \subseteq \{1,\ldots,p_n\} \setminus \{i,j\}$.

Many partial correlations (and non-partial correlations) are tested for being zero during the run of the PC(m_n)-algorithm. For a fixed ordered pair of nodes i,j, the conditioning sets are elements of

$$K_{i,j}^{m_n} = \{\mathbf{k} \subseteq \{1,\ldots,p_n\} \setminus \{i,j\} : |\mathbf{k}| \leq m_n\}$$

whose cardinality is bounded by

$$|K_{i,j}^{m_n}| \leq B p_n^{m_n} \text{ for some } 0 < B < \infty. \tag{5}$$

Lemma 1 *Assume (A1) (without requiring faithfulness) and* $\sup_{n,i\neq j}|\rho_{n;i,j}| \leq M < 1$ *(compare with (A4)). Then, for any* $0 < \gamma \leq 2$*,*

$$\sup_{i,j,\mathbf{k}\in K_{i,j}^{m_n}} \mathbf{P}[|\hat{\rho}_{n;i,j} - \rho_{n;i,j}| > \gamma] \leq C_1(n-2)\exp\left((n-4)\log(\frac{4-\gamma^2}{4+\gamma^2})\right),$$

for some constant $0 < C_1 < \infty$ *depending on M only.*

Proof: We make substantial use of Hotelling (1953)'s work. Denote by $f_n(\hat{\rho}, \rho)$ the probability density function of the sample correlation $\hat{\rho} = \hat{\rho}_{n+1;i,j}$ based on $n+1$ observations and by $\rho = \rho_{n+1;i,j}$ the population correlation. (It is notationally easier to work with sample size $n+1$; and we just use the abbreviated notations with $\hat{\rho}$ and ρ). For $0 < \gamma \leq 2$,

$$\mathbf{P}[|\hat{\rho} - \rho| > \gamma] = \mathbf{P}[\hat{\rho} < \rho - \gamma] + \mathbf{P}[\hat{\rho} > \rho + \gamma].$$

It can be shown, that $f_n(r, \rho) = f_n(-r, -\rho)$, see Hotelling (1953, p.201). This symmetry implies,

$$\mathbf{P}_{\rho}[\hat{\rho} < \rho - \gamma] = \mathbf{P}_{\tilde{\rho}}[\hat{\rho} > \tilde{\rho} + \gamma] \text{ with } \tilde{\rho} = -\rho. \tag{6}$$

Thus, it suffices to show that $\mathbf{P}[\hat{\rho} > \rho + \gamma] = \mathbf{P}_{\rho}[\hat{\rho} > \rho + \gamma]$ decays exponentially in n, uniformly for all ρ.

It has been shown (Hotelling, 1953, p.201, Formula (29)), that for $-1 < \rho < 1$,

$$\mathbf{P}[\hat{\rho} > \rho + \gamma] \leq \frac{(n-1)\Gamma(n)}{\sqrt{2\pi}\Gamma(n+\frac{1}{2})} M_0(\rho+\gamma)(1 + \frac{2}{1-|\rho|}) \tag{7}$$

with

$$\begin{aligned} M_0(\rho+\gamma) &= \int_{\rho+\gamma}^{1} (1-\rho^2)^{\frac{n}{2}} (1-x^2)^{\frac{n-3}{2}} (1-\rho x)^{-n+\frac{1}{2}} dx \\ &= \int_{\rho+\gamma}^{1} (1-\rho^2)^{\frac{\tilde{n}+3}{2}} (1-x^2)^{\frac{\tilde{n}}{2}} (1-\rho x)^{-\tilde{n}-\frac{5}{2}} dx \ \ (\text{using } \tilde{n} = n-3) \\ &\leq \frac{(1-\rho^2)^{\frac{3}{2}}}{(1-|\rho|)^{\frac{5}{2}}} \int_{\rho+\gamma}^{1} (\frac{\sqrt{1-\rho^2}\sqrt{1-x^2}}{1-\rho x})^{\tilde{n}} dx \\ &\leq \frac{(1-\rho^2)^{\frac{3}{2}}}{(1-|\rho|)^{\frac{5}{2}}} 2 \max_{\rho+\gamma\leq x\leq 1} (\frac{\sqrt{1-\rho^2}\sqrt{1-x^2}}{1-\rho x})^{\tilde{n}}. \end{aligned} \tag{8}$$

We will show now that $g_{\rho}(x) = \frac{\sqrt{1-\rho^2}\sqrt{1-x^2}}{1-\rho x} < 1$ for all $\rho + \gamma \leq x \leq 1$ and $-1 < \rho < 1$ (in fact, $\rho \leq 1-\gamma$ due to the first restriction). Consider

$$\begin{aligned} \sup_{-1<\rho<1;\rho+\gamma\leq x\leq 1} g_{\rho}(x) &= \sup_{-1<\rho\leq 1-\gamma} \frac{\sqrt{1-\rho^2}\sqrt{1-(\rho+\gamma)^2}}{1-\rho(\rho+\gamma)} \\ &= \frac{\sqrt{1-\frac{\gamma^2}{4}}\sqrt{1-\frac{\gamma^2}{4}}}{1-(\frac{-\gamma}{2})(\frac{\gamma}{2})} = \frac{4-\gamma^2}{4+\gamma^2} < 1 \text{ for all } 0 < \gamma \leq 2. \end{aligned} \tag{9}$$

Therefore, for $-1 < -M \leq \rho \leq M < 1$ (see assumption (A4)) and using Formulas (7)-(9) together with the fact that $\frac{\Gamma(n)}{\Gamma(n+\frac{1}{2})} \leq const.$ with respect to n, we have

$$\begin{aligned} & \mathbf{P}[\hat{\rho} > \rho + \gamma] \\ \leq \quad & \frac{(n-1)\Gamma(n)}{\sqrt{2\pi}\Gamma(n+\frac{1}{2})} \frac{(1-\rho^2)^{\frac{3}{2}}}{(1-|\rho|)^{\frac{5}{2}}} 2(\frac{4-\gamma^2}{4+\gamma^2})^{\tilde{n}}(1+\frac{2}{1-|\rho|}) \\ \leq \quad & \frac{(n-1)\Gamma(n)}{\sqrt{2\pi}\Gamma(n+\frac{1}{2})} \frac{1}{(1-M)^{\frac{5}{2}}} 2(\frac{4-\gamma^2}{4+\gamma^2})^{\tilde{n}}(1+\frac{2}{1-M}) \leq \\ \leq \quad & C_1(n-1)(\frac{4-\gamma^2}{4+\gamma^2})^{\tilde{n}} = C_1(n-1)\exp((n-3)\log(\frac{4-\gamma^2}{4+\gamma^2})), \end{aligned}$$

where $0 < C_1 < \infty$ depends on M only, but not on ρ or γ. By invoking Formula (6), the proof is complete (note that the proof assumed sample size $n+1$). □

Lemma 1 can be easily extended to partial correlations, as shown by Fisher (1924), using projections for Gaussian distributions.

Lemma 2 *(Fisher, 1924)*
Assume (A1) (without requiring faithfulness). If the cumulative distribution function of $\hat{\rho}_{n;i,j}$ is denoted by $F(\cdot|n, \rho_{n;i,j})$, then the cdf of the sample partial correlation $\hat{\rho}_{n;i,j|\mathbf{k}}$ with $|\mathbf{k}| = m < n-1$ is $F[\cdot|n-m, \rho_{n;i,j|\mathbf{k}}]$. That is, the effective sample size is reduced by m.

A proof can be found in Fisher (1924); see also Anderson (1984). □

Lemma 1 and 2 yield then the following.

Corollary 1 *Assume (the first part of) (A1) and (the upper bound in) (A4). Then, for any $\gamma > 0$,*

$$\begin{aligned} & \sup_{i,j,\mathbf{k} \in K_{i,j}^{m_n}} \mathbf{P}[|\hat{\rho}_{n;i,j|\mathbf{k}} - \rho_{n;i,j|\mathbf{k}}| > \gamma] \\ \leq \quad & C_1(n-2-m_n)\exp\left((n-4-m_n)\log(\frac{4-\gamma^2}{4+\gamma^2})\right), \end{aligned}$$

for some constant $0 < C_1 < \infty$ depending on M from (A4) only.

The PC-algorithm is testing partial correlations after the z-transform $g(\rho) = 0.5\log((1+\rho)/(1-\rho))$. Denote by $Z_{n;i,j|\mathbf{k}} = g(\hat{\rho}_{n;i,j|\mathbf{k}})$ and by $z_{n;i,j|\mathbf{k}} = g(\rho_{n;i,j|\mathbf{k}})$.

Lemma 3 *Assume the conditions from Corollary 1. Then, for any $\gamma > 0$,*

$$\begin{aligned} & \sup_{i,j,\mathbf{k} \in K_{i,j}^{m_n}} \mathbf{P}[|Z_{n;i,j|\mathbf{k}} - z_{n;i,j|\mathbf{k}}| > \gamma] \\ \leq \quad & O(n-m_n)\left(\exp((n-4-m_n)\log(\frac{4-(\gamma/L)^2}{4+(\gamma/L)^2})) + \exp(-C_2(n-m_n))\right) \end{aligned}$$

for some constant $0 < C_2 < \infty$ and $L = 1/(1-(1+M)^2/4)$.

Proof: A Taylor expansion of the z-transform $g(\rho) = 0.5\log((1+\rho)/(1-\rho))$ yields:

$$Z_{n;i,j|\mathbf{k}} - z_{n;i,j|\mathbf{k}} = g'(\tilde{\rho}_{n;i,j|\mathbf{k}})(\hat{\rho}_{n;i,j|\mathbf{k}} - \rho_{n;i,j|\mathbf{k}}), \tag{10}$$

where $|\tilde{\rho}_{n;i,j|\mathbf{k}} - \rho_{n;i,j|\mathbf{k}}| \leq |\hat{\rho}_{n;i,j|\mathbf{k}} - \rho_{n;i,j|\mathbf{k}}|$. Moreover, $g'(\rho) = 1/(1-\rho^2)$. By applying Corollary 1 with $\gamma = \kappa = (1-M)/2$ we have

$$\begin{aligned} &\sup_{i,j,\mathbf{k}\in K_{i,j}^{m_n}} \mathbf{P}[|\tilde{\rho}_{n;i,j|\mathbf{k}} - \rho_{n;i,j|\mathbf{k}}| \leq \kappa] \\ > \ & 1 - C_1(n-2-m_n)\exp(-C_2(n-m_n)). \end{aligned} \tag{11}$$

Since

$$\begin{aligned} g'(\tilde{\rho}_{n;i,j|\mathbf{k}}) &= \frac{1}{1-\tilde{\rho}_{n;i,j|\mathbf{k}}^2} = \frac{1}{1-(\rho_{n;i,j|\mathbf{k}} + (\tilde{\rho}_{n;i,j|\mathbf{k}} - \rho_{n;i,j|\mathbf{k}}))^2} \\ &\leq \frac{1}{1-(M+\kappa)^2} \text{ if } |\tilde{\rho}_{n;i,j|\mathbf{k}} - \rho_{n;i,j|\mathbf{k}}| \leq \kappa, \end{aligned}$$

where we also invoke (the second part of) assumption (A4) for the last inequality. Therefore, since $\kappa = (1-M)/2$ yielding $1/(1-(M+\kappa)^2) = L$, and using Formula (11), we get

$$\begin{aligned} &\sup_{i,j,\mathbf{k}\in K_{i,j}^{m_n}} \mathbf{P}[|g'(\tilde{\rho}_{n;i,j|\mathbf{k}})| \leq L] \\ \geq \ & 1 - C_1(n-2-m_n)\exp(-C_2(n-m_n)). \end{aligned} \tag{12}$$

Since $|g'(\rho)| \geq 1$ for all ρ, we obtain with Formula (10):

$$\begin{aligned} &\sup_{i,j,\mathbf{k}\in K_{i,j}^{m_n}} \mathbf{P}[|Z_{n;i,j|\mathbf{k}} - z_{n;i,j|\mathbf{k}}| > \gamma] \\ \leq \ & \sup_{i,j,\mathbf{k}\in K_{i,j}^{m_n}} \mathbf{P}[|g'(\tilde{\rho}_{n;i,j|\mathbf{k}})| > L] + \sup_{i,j,\mathbf{k}\in K_{i,j}^{m_n}} \mathbf{P}[|\hat{\rho}_{n;i,j|\mathbf{k}} - \rho_{n;i,j|\mathbf{k}}| > \gamma/L]. \end{aligned} \tag{13}$$

Formula (13) follows from elementary probability calculations: for two random variables U, V with $|U| \geq 1$ ($|U|$ corresponding to $|g'(\tilde{\rho})|$ and $|V|$ to the difference $|\hat{\rho} - \rho|$),

$$\begin{aligned} \mathbf{P}[|UV| > \gamma] &= \mathbf{P}[|UV| > \gamma, |U| > L] + \mathbf{P}[|UV| > \gamma, 1 \leq |U| \leq L] \\ &\leq \mathbf{P}[|U| > L] + \mathbf{P}[|V| > \gamma/L]. \end{aligned}$$

The statement then follows from Formulas (13), (12) and Corollary 1. □

A.2.2 Proof of Theorem 1

For the analysis of the PC-algorithm, it is useful to consider a more general version as shown in Algorithm 3.

The PC-algorithm in Section 2.2.1 equals the PC$_{pop}$(m_{reach})-algorithm . There is the obvious sample version, the PC(m)-algorithm, and the PC-algorithm in Section 2.2.2 is then same as the PC($\hat{m}_{reach}$)-algorithm, where $\hat{m}_{reach}$ is the sample version of Formula (2).

The population version PC$_{pop}$(m_n)-algorithm when stopped at level $m_n = m_{reach,n}$ constructs the true skeleton according to Proposition 1. Moreover, the PC$_{pop}$(m)-algorithm remains to be correct when using $m \geq m_{reach,n}$. The following Lemma extends this result to the sample PC(m)-algorithm.

Algorithm 3 The PC$_{pop}(m)$-algorithm

```
INPUT: Stopping level $m$, Vertex Set $V$, Conditional Independence Information
OUTPUT: Estimated skeleton $C$, separation sets $S$ (only needed when directing the skeleton afterwards)
Form the complete undirected graph $\tilde{C}$ on the vertex set V.
$\ell = -1; \quad C = \tilde{C}$
repeat
  $\ell = \ell + 1$
  repeat
    Select a (new) ordered pair of nodes $i$,$j$ that are adjacent in $C$ such that $|adj(C,i) \setminus \{j\}| \geq \ell$
    repeat
      Choose (new) $\mathbf{k} \subseteq adj(C,i) \setminus \{j\}$ with $|\mathbf{k}| = \ell$.
      if $i$ and $j$ are conditionally independent given $\mathbf{k}$ then
        Delete edge $i, j$
        Denote this new graph by $C$.
        Save $\mathbf{k}$ in $S(i,j)$ and $S(j,i)$
      end if
    until edge $i, j$ is deleted or all $\mathbf{k} \subseteq adj(C,i) \setminus \{j\}$ with $|\mathbf{k}| = \ell$ have been chosen
  until all ordered pairs of adjacent variables $i$ and $j$ such that $|adj(C,i) \setminus \{j\}| \geq \ell$ and $\mathbf{k} \subseteq adj(C,i) \setminus \{j\}$ with $|\mathbf{k}| = \ell$ have been tested for conditional independence
until $\ell = m$ or for each ordered pair of adjacent nodes $i$,$j$: $|adj(C,i) \setminus \{j\}| < \ell$.
```

Lemma 4 *Assume (A1), (A2), (A3) where* $0 < b \leq 1$ *and (A4) where* $0 < d < b/2$*. Denote by* $\hat{G}_{skel,n}(\alpha_n, m_n)$ *the estimate from the PC(m_n)-algorithm in Section 2.2.2 and by* $G_{skel,n}$ *the true skeleton from the DAG* G_n*. Moreover, denote by* $m_{reach,n}$ *the value described in (2). Then, for* $m_n \geq m_{reach,n}$, $m_n = O(n^{1-b})$ $(n \to \infty)$*, there exists* $\alpha_n \to 0$ $(n \to \infty)$ *such that*

$$\begin{aligned} &\mathbf{P}[\hat{G}_{skel,n}(\alpha_n, m_n) = G_{skel,n}] \\ = \quad & 1 - O(\exp(-Cn^{1-2d})) \to 1 \ (n \to \infty) \textit{ for some } 0 < C < \infty. \end{aligned}$$

Proof: An error occurs in the sample PC-algorithm if there is a pair of nodes i, j and a conditioning set $\mathbf{k} \in K_{i,j}^{m_n}$ (although the algorithm is typically only going through a random subset of $K_{i,j}^{m_n}$) where an error event $E_{i,j|\mathbf{k}}$ occurs; $E_{i,j,\mathbf{k}}$ denotes that "an error occurred when testing partial correlation for zero at nodes i, j with conditioning set $\mathbf{k}$". Thus,

$$\begin{aligned} &\mathbf{P}[\text{an error occurs in the PC}(m_n)\text{-algorithm}] \\ \leq \quad & P[\bigcup_{i,j,\mathbf{k} \in K_{ij}^{m_n}} E_{i,j|\mathbf{k}}] \leq O(p_n^{m_n+2}) \sup_{i,j,\mathbf{k} \in K_{ij}^{m_n}} \mathbf{P}[E_{i,j|\mathbf{k}}], \end{aligned} \tag{14}$$

using that the cardinality of the set $|\{i, j, \mathbf{k} \in K_{ij}^{m_n}\}| = O(p_n^{m_n+2})$, see also Formula 5. Now

$$E_{i,j|\mathbf{k}} = E_{i,j|\mathbf{k}}^{I} \cup E_{i,j|\mathbf{k}}^{II}, \tag{15}$$

where

$$\begin{aligned} &\text{type I error } E_{i,j|\mathbf{k}}^{I}: \ \sqrt{n - |k| - 3}|Z_{i,j|\mathbf{k}}| > \Phi^{-1}(1 - \alpha/2) \text{ and } z_{i,j|\mathbf{k}} = 0, \\ &\text{type II error } E_{i,j|\mathbf{k}}^{II}: \ \sqrt{n - |k| - 3}|Z_{i,j|\mathbf{k}}| \leq \Phi^{-1}(1 - \alpha/2) \text{ and } z_{i,j|\mathbf{k}} \neq 0. \end{aligned}$$

Choose $\alpha = \alpha_n = 2(1 - \Phi(n^{1/2}c_n/2))$, where c_n is from (A4). Then,

$$\begin{aligned}\sup_{i,j,\mathbf{k}\in K_{i,j}^{m_n}} \mathbf{P}[E_{i,j|\mathbf{k}}^{I}] &= \sup_{i,j,\mathbf{k}\in K_{i,j}^{m_n}} \mathbf{P}[|Z_{i,j|\mathbf{k}} - z_{i,j|\mathbf{k}}| > (n/(n-|\mathbf{k}|-3))^{1/2}c_n/2] \\ &\leq O(n-m_n)\exp(-C_3(n-m_n)c_n^2), \qquad (16)\end{aligned}$$

for some $0 < C_3 < \infty$ using Lemma 3 and the fact that $\log(\frac{4-\delta^2}{4+\delta^2}) \sim -\delta^2/2$ as $\delta \to 0$. Furthermore, with the choice of $\alpha = \alpha_n$ above,

$$\begin{aligned}&\sup_{i,j,\mathbf{k}\in K_{i,j}^{m_n}} \mathbf{P}[E_{i,j|\mathbf{k}}^{II}] = \sup_{i,j,\mathbf{k}\in K_{i,j}^{m_n}} \mathbf{P}[|Z_{i,j|\mathbf{k}}| \leq \sqrt{n/(n-|\mathbf{k}|-3)}c_n/2] \\ \leq\ &\sup_{i,j,\mathbf{k}\in K_{i,j}^{m_n}} \mathbf{P}[|Z_{i,j|\mathbf{k}} - z_{i,j|\mathbf{k}}| > c_n(1-\sqrt{n/(n-|\mathbf{k}|-3)}/2)],\end{aligned}$$

because $\inf_{i,j;\mathbf{k}\in K_{i,j}^{m_n}} |z_{i,j|\mathbf{k}}| \geq c_n$ since $|g(\rho)| \geq |\rho|$ for all ρ and using assumption (A4). By invoking Lemma 3 we then obtain:

$$\sup_{i,j,\mathbf{k}\in K_{i,j}^{m_n}} \mathbf{P}[E_{i,j|\mathbf{k}}^{II}] \leq O(n-m_n)\exp(-C_4(n-m_n)c_n^2) \qquad (17)$$

for some $0 < C_4 < \infty$. Now, by Formulas (14)-(17) we get

$$\begin{aligned}&\mathbf{P}[\text{an error occurs in the PC}(m_n)\text{-algorithm}] \\ \leq\ &O(p_n^{m_n+2}(n-m_m)\exp(-C_5(n-m_n)c_n^2)) \\ \leq\ &O(n^{a(m_n+2)+1}\exp(-C_5(n-m_n)n^{-2d})) \\ =\ &O\Big(\exp\Big(a(m_n+2)\log(n)+\log(n)-C_5(n^{1-2d}-m_nn^{-2d})\Big)\Big) = o(1),\end{aligned}$$

because n^{1-2d} dominates all other terms in the argument of the exp-function due to the assumption in (A4) that $d < b/2$. This completes the proof. □

Lemma 4 leaves some flexibility for choosing m_n. The PC-algorithm yields a data-dependent reached stopping level $\hat{m}_{reach,n}$, that is, the sample version of (2).

Lemma 5 *Assume (A1)-(A4). Then,*

$$\begin{aligned}&\mathbf{P}[\hat{m}_{reach,n} = m_{reach,n}] = 1 - O(\exp(-Cn^{1-2d})) \to 1\ (n \to \infty) \\ &\textit{for some } 0 < C < \infty,\end{aligned}$$

where $d > 0$ is as in (A4).

Proof: Consider the population algorithm PC$_{pop}(m)$: the reached stopping level satisfies $m_{reach} \in \{q_n - 1, q_n\}$, see Proposition 1. The sample PC(m_n)-algorithm with stopping level in the range of $m_{reach} \leq m_n = O(n^{1-b})$, coincides with the population version on a set A having probability $P[A] = 1 - O(\exp(-Cn^{1-2d}))$, see the last formula in the proof of Lemma 4. Hence, on the set A, $\hat{m}_{reach,n} = m_{reach} \in \{q_n - 1, q_n\}$. The claim then follows from Lemma 4. □

Lemma 4 and 5 together complete the proof of Theorem 1.

Because there are faithful distributions which require $m_n = m_{reach,n} \in \{q_n - 1, q_n\}$ for consistent estimation with the PC(m)-algorithm, Lemma 5 indicates that the PC-algorithm, stopping at $\hat{m}_{reach,n}$, yields with high probability the smallest $m = m_n$ which is universally consistent for all faithful distributions.

A.2.3 PROOF OF THEOREM 2

As mentioned in Section 2.3, due to the result of Meek (1995b), it is sufficient to estimate the correct skeleton and separation sets. The proof of Theorem 1 also covers the issue of choosing the correct separation sets S, that is, the probability of estimating the correct sets S goes to one as $n \to \infty$. Hence, the proof of Theorem 2 is completed.

Appendix B. Bound for error probability of PC-algorithm

M and c are the upper and lower bounds for partial correlations, as defined in Section 3.1. p is the number of variables, q is the maximal size of neighbors, n is the sample size. The significance level is chosen as suggested in the proofs, that is, $\alpha = 2(1-\Phi(n^{1/2}c/2))$. By closely inspecting the proofs, one can derive the following upper bound for the error probability of the PC-algorithm:

$$\mathbf{P}[\hat{G} \neq G] \leq p^{q+2} C_1 (n-1-q)(\exp(-C_2(n-q)) + \exp((n-4-q) f(L, \frac{c}{2})))$$

where $L = \frac{1}{1-(1+M)^2/4}$, $C_1 = \frac{1+2/(1-M)}{(1-M)^{5/2}}$, $C_2 = -\log(\frac{16-(1-M)^2}{16+(1-M)^2})$ and $f(x,y) = \log(\frac{4-(y/x)^2}{4+(y/x)^2})$.

References

T.W. Anderson. *An Introduction to Multivariate Statistical Analysis*. Wiley, 2nd edition, 1984.

D.M. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2002a.

D.M. Chickering. Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research*, 2:445–498, 2002b.

C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

D. Edwards. *Introduction to Graphical Modelling*. Springer Verlag, 2nd edition edition, 2000.

R.A. Fisher. The distribution of the partial correlation coefficient. *Metron*, 3:329–332, 1924.

S.B. Gillispie and M.D. Perlman. Enumerating Markov equivalence classes of acyclic digraph models. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, pages 171–177, 2001.

A. Goldenberg and A. Moore. Tractable learning of large bayes net structures from sparse data. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, pages 44–51. ACM Press, 2004.

D. Heckerman, D. Geiger, and D.M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

H. Hotelling. New light on the correlation coefficient and its transforms. *Journal of the Royal Statistical Society Series B*, 15(2):193–232, 1953.

S. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

C. Meek. Strong completeness and faithfulness in Bayesian networks. In *Uncertainty in Artificial Intelligence*, pages 411–418, 1995a.

C. Meek. Causal inference and causal explanation with background knowledge. In P.Besnard and S.Hanks, editors, *Uncertainty in Artificial Intelligence*, volume 11, pages 403–410, 1995b.

N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the Lasso. *Annals of Statistics*, 34:1436–1462, 2006.

R.E. Neapolitan. *Learning Bayesian Networks*. Pearson Prenctice Hall, 2004.

A. Y. Ng. On feature selection: Learning with exponentially many irrelevant features as training examples. In *Proc. 15th International Conf. on Machine Learning*, pages 404–412. Morgan Kaufmann, San Francisco, CA, 1998.

J. Pearl. *Causality*. Cambridge University Press, 2000.

J.M. Robins, R. Scheines, P. Sprites, and L. Wasserman. Uniform consistency in causal inference. *Biometrika*, 90:491–515, 2003.

R.W. Robinson. Counting labeled acyclic digraphs. In F. Haray, editor, *New Directions in the Theory of Graphs: Proc. of the Third Ann Arbor Conf. on Graph Theory (1971)*, pages 239–273. Academic Press, NY, 1973.

D.J. Spiegelhalter, A.P. Dawid, S.L. Lauritzen, and R.G. Cowell. Bayesian analysis in expert-systems (with discussion). *Statistical Science*, 8:219–283, 1993.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. The MIT Press, 2nd edition, 2000.

I. Tsamardinos, L.E. Brown, and C.F. Aliferis. The max-min hill-climbing bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.

T. Verma and J.Pearl. A theory of inferred causation. In J. Allen, R. Fikes, and E. Sandewall, editors, *Knowledge Representation and Reasoning: Proceedings of the Second International Conference*, pages 441–452. Morgan Kaufmann, New York, 1991.

T. Verma and J. Pearl. Equivalence and synthesis of causal models. In M. Henrion, M. Shachter, R. Kanal, and J. Lemmer, editors, *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, pages 220–227, 1990.

J. Zhang and P. Spirtes. Strong faithfulness and uniform consistency in causal inference. In *UAI*, pages 632–639, 2003.

P. Zhao and B. Yu. On model selection consistency of Lasso. *Journal of Machine Learning Research*, 7:2541–2563, 2006.

O. Zuk, S. Margel, and E. Domany. On the number of samples needed to learn the correct structure of a Bayesian network. In *UAI*, 2006.

Journal of Machine Learning Research 8 (2007) 637-652 Submitted 2/06; Revised 10/06; Published 3/07

Margin Trees for High-dimensional Classification

Robert Tibshirani TIBS@STAT.STANFORD.EDU
Trevor Hastie HASTIE@STAT.STANFORD.EDU
Departments of Health, Research & Policy, and Statistics
Stanford University
Stanford, CA 94305

Editor: Dale Schurmanns

Abstract

We propose a method for the classification of more than two classes, from high-dimensional features. Our approach is to build a binary decision tree in a top-down manner, using the optimal margin classifier at each split. We implement an exact greedy algorithm for this task, and compare its performance to less greedy procedures based on clustering of the matrix of pairwise margins. We compare the performance of the "margin tree" to the closely related "all-pairs" (one versus one) support vector machine, and nearest centroids on a number of cancer microarray data sets. We also develop a simple method for feature selection. We find that the margin tree has accuracy that is competitive with other methods and offers additional interpretability in its putative grouping of the classes.

Keywords: maximum margin classifier, support vector machine, decision tree, CART

1. Introduction

We consider the problem of classifying objects into two or more classes, from a set of features. Our main application area is the classification of cancer patient samples from gene expression measurements.

When the number of classes K is greater than two, maximum margin classifiers do not generalize easily. Various approaches have been suggested, some based on the two-class classifier ("one-versus-all" and "one-versus one' or "all pairs'), and others modifying the support vector loss function to deal directly with more than two classes (Weston and Watkins, 1999; Lee et al., 2004; Rosset et al., 2005). These latter proposals have a nice generalization of the maximum margin property of the two class support vector classifier. Statnikov et al. (2004) contains a comparison of different support vector approaches to classification from microarray gene expression cancer data sets. While these methods can produce accurate predictions, they lack interpretability. In particular, with a large number of classes, the investigator may want not only a classifier but also a meaningful organization of the classes.

In this paper we propose a tree-based maximum margin classifier. Figure 1 illustrates our idea. There are three classes and two features, as shown in the top left panel. We seek the line that partitions the classes into two groups, that has the maximum margin. (The margin is the minimum distance to the decision line among all of the data points.)

The best line is shown in the top right panel, splitting class 1 from classes 2 and 3. We then focus just on classes 2 and 3, and their maximum margin classifier is shown in the bottom left. The overall top-down classifier is summarized by the binary tree shown in the bottom right panel.

We employ strategies like this for larger numbers of classes, producing a binary decision tree with a maximum margin classifier at each junction in the tree.

In Section 2 we give details of the margin tree classifier. Section 3 shows the application of the margin tree to a number of cancer microarray data sets. For construction of the tree, all of the classifiers in the margin tree use all of the features (genes). In Section 4 we discuss approaches to feature selection. Finally in Section 5 we have some further comments and a discussion of related work in the literature.

2. The Margin Tree Classifier

Denote the gene expression profiles by $x_j = (x_{1j}, x_{2j}, \dots x_{pj})$ for $j = 1, 2, \dots N$ samples falling into one of K classes. The features (genes) are indexed by $i = 1, 2, \dots p$.

Consider first the case of $K = 2$ classes, C_1 and C_2. The class outcome is denoted by $y_j = \pm 1$. The maximum margin classifier is defined by the constant β_0 and the weight vector β with components $\sum_i \beta_i^2 = 1$ that maximizes the gap between the classes, or the "margin". Formally,

$$\begin{aligned}(\beta_0, \beta) &= \text{argmax}(C) \\ &\quad \text{where } y_j(\beta_0 + \sum_i \beta_i x_{ij}) \geq C \ \forall j.\end{aligned}$$

The achieved margin $M = 2 \cdot C$. In the examples of this paper, $p > N$ so that all classes are separable and $M > 0$. We have some discussion of the non-separable case in Section 5.

Now suppose we have $K > 2$ classes. We consider three different strategies for constructing the tree. These use different criteria for deciding on the best partition of the classes into two groups at each juncture. Having settled on the partition, we use the maximum margin classifier between the two groups of classes, for future predictions.

Let $M(j,k)$ be the maximum margin between classes j and k. Also, let G_1, G_2 be groups of classes, and let $M(G_1, G_2)$ denote the maximum margin between the groups. That is, $M(G_1, G_2)$ is the maximum margin between two hyper-classes: all classes in G_1 and all classes in G_2. Finally, denote a partition by $P = \{G_1, G_2\}$.

Then we consider three approaches for splitting a node in the decision tree:

(a) *Greedy:* maximize $M(G_1, G_2)$ over all partitions P.

(b) *Single linkage:* Find the partition P yielding the largest margin M_0 so that min $M(j_1, j_2) \leq M_0$ for $j_1, j_2 \in G_k, k = 1, 2$ and min $M(j_1, j_2) \geq M$ for $j_1 \in G_1, j_2 \in G_2$.

(c) *Complete linkage:* Find the partition P yielding the largest margin M_0 so that max $M(j_1, j_2) \leq M_0$ for $j_1, j_2 \in G_k, k = 1, 2$ and max $M(j_1, j_2) \geq M_0$ for $j_1 \in G_1, j_2 \in G_2$.

The greedy method finds the partition that maximizes the resulting margin over all possible partitions. Although this may seem prohibitive to compute for a large number of classes, we derive an exact, reasonably fast algorithm for this approach (details below).

The second and third methods require some explanation. They are derived from the *bottom up* (as opposed to top-down) clustering methods. Each one requires just the $\binom{K}{2}$ margin classifiers for

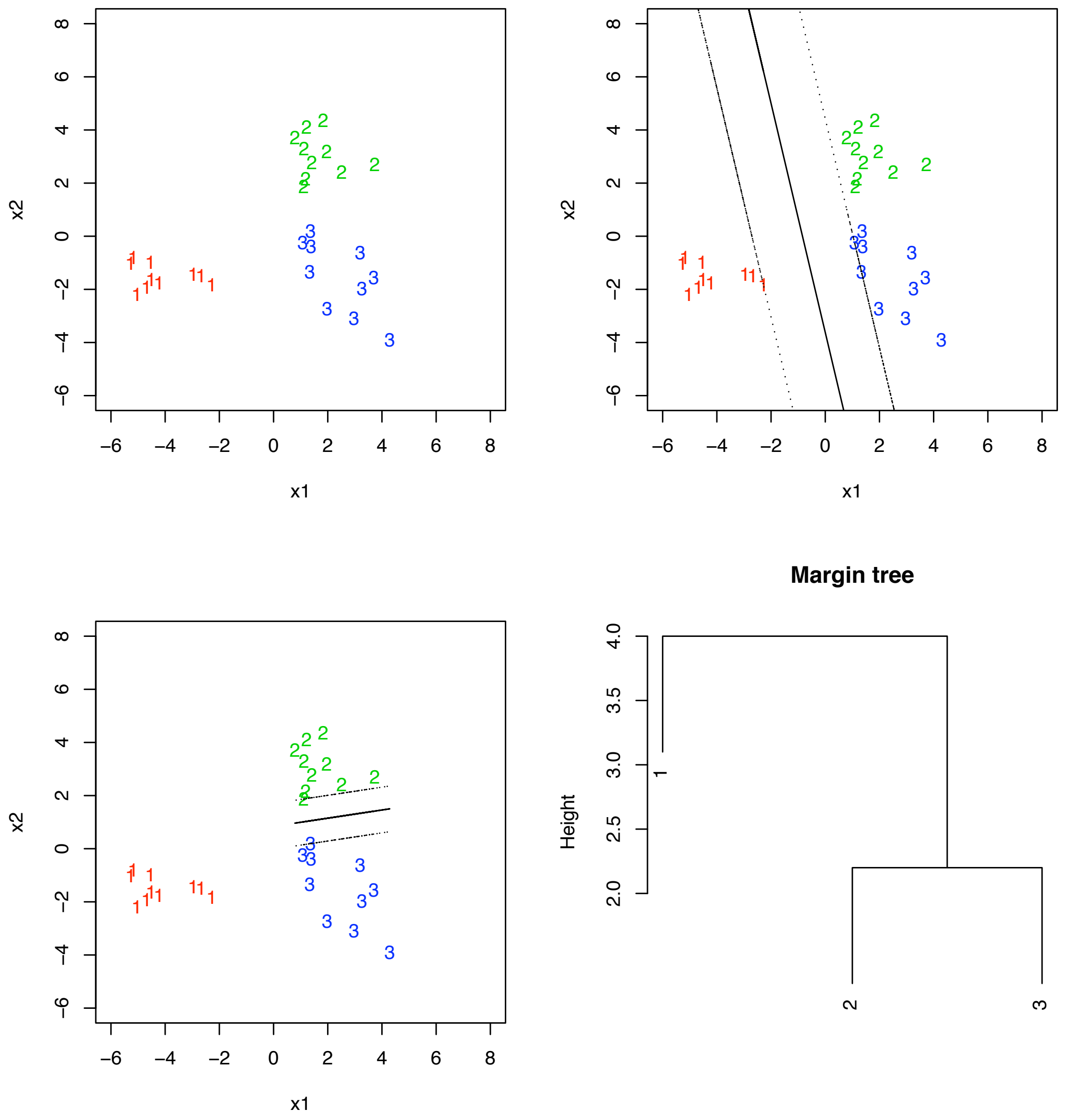


Figure 1: Simple illustration of a margin tree. There are three classes shown in the top left panel. The largest margin is between class 1 and (2,3), with the optimal classifier shown on the top right. Then we separate class 2 from 3, in the bottom left. These top-down splits are summarized in the margin tree in the bottom right.

each pair of classes. Single linkage clustering successively merges groups based on the minimum distance between any pair of items in each of the group. Complete linkage clustering does the same, but using the maximum distance. Now having built a clustering tree bottom-up, we can interpret each split in the tree in a *top-down* manner, and that is how criteria (b) and (c) above were derived. In particular it is easy to see that the single and complete linkage problems are solved by single and complete linkage agglomerative clustering, respectively, applied to the margin matrix $M(j_1, j_2)$. Note that we are applying single or complete linkage clustering to the classes of objects C_j, while one usually applies clustering to individual objects.

The greedy method focuses on the form of the final classifier, and tries to optimize that classification at each stage. Note that the greedy method cares only about the distance between classes in the different partitions, and not about the distance between classes within the same partition. Both the single linkage and complete linkage methods take into account both the between and within partition distances. We will also see in the next section that the complete linkage method can be viewed as an approximation to the greedy search.

Figure 2 shows a toy example that illustrates the difference between the greedy and complete linkage algorithms. There are six classes with circular distributions. The greedy algorithm splits off group 1,2, and 3 in succession, and then splits off 4,5,6 as a group. This is summarized in the bottom left panel. The complete linkage algorithm in the bottom right panel instead groups 1,2 and 3 together and 4,5, and 6 together. The complete linkage tree is more balanced and hence may be more useful biologically.

In the experiments in this paper we find that:

- All three methods produce about the same test set accuracy, and about the same as the all-pairs maximum margin classifier.

- The complete linkage approach gives more balanced trees, that may be more interpretable that those from the other two methods; the single linkage and greedy methods tend to produce long stringy trees that usually split off one class at a time at each branch. The complete linkage method is also considerably faster to compute than the greedy method.

Thus the complete linkage margin tree emerges as our method of choice. It requires computation of $\binom{K}{2}$ support vector classifiers for each pair of classes for the complete linkage clustering and then for the final tree, one computation of a support vector classifier for each node in the tree (at most K and typically $\approx \log_2(K)$ classifiers.)

2.1 An Exact Algorithm for the Greedy Criterion

A key fact is

$$M(G_1, G_2) \leq \min\{M(j_1, j_2), j_1 \in G_1, j_2 \in G_2\}. \tag{1}$$

That is, the margin between two groups of classes is less than or equal to the smallest margin between any pair of classes, one chosen from each group.

Now suppose we seek a partition P with margin M. Rather than enumerate all possible partitions (and their associated maximum margin classifiers), we can speed up the computation by constructing the complete linkage clustering tree, and collapsing all nodes at height M. We know that all classes in any collapsed node must be on the same side of the decision plane, since each class has margin

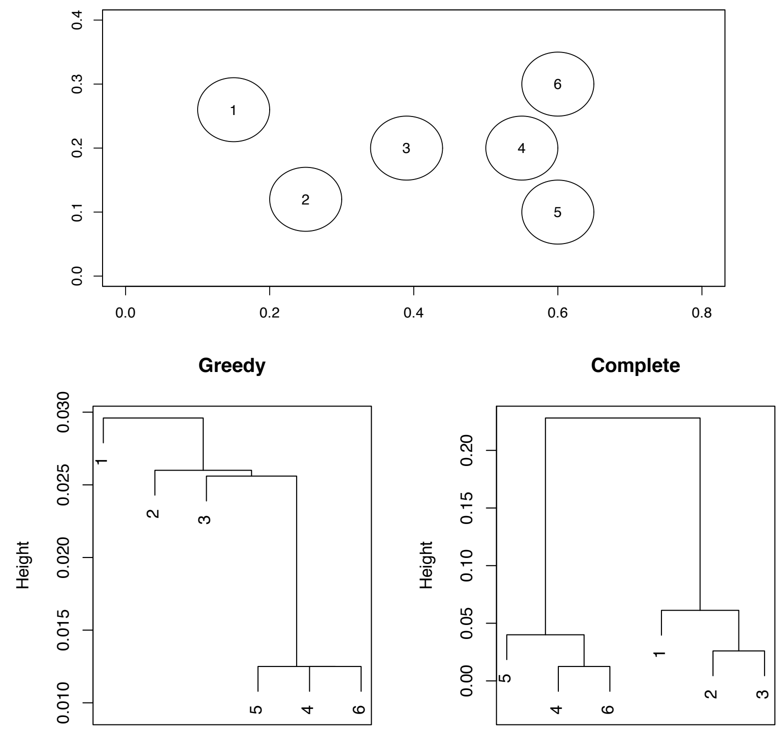


Figure 2: A toy example illustrating the difference between the greedy and complete linkage algorithms. There are six classes with circular distributions (top panel). The greedy algorithm splits off groups 1,2, and 3 in succession, and then splits off 4,5,6 as a group. This is summarized in the bottom left panel. The complete linkage algorithm (bottom right panel) instead groups 1,2 and 3 together, and 4,5, and 6 together. For example, the margin between classes 1 and 2 is 0.0296, while that between 3 and 4 is less: 0.0256. The height in each plot is the margin corresponding to each join.

at least M with every other class in that node. Hence we need only consider partitions that keep the collapsed nodes intact.

We summarize the algorithm below:

Exact computation of the best greedy split

1. Construct the complete linkage clustering tree based on the margin matrix $M(j_1, j_2)$.
2. Starting with all classes at the top of the tree, find the partition of each individual class versus the rest, and also the partition that produces two classes in the complete linkage tree (that is, make a horizontal cut in the tree to produce two classes). Let M_0 be the largest margin achieved amongst all of these competitors.

3. Cut the complete linkage tree at height M_0, and collapse all nodes at that height.

4. Consider all partitions of all classes that keep the collapsed nodes intact, and choose the one that gives maximal margin $\hat{M}$.

This procedure finds the partition of the classes that yields the maximum margin. We then apply this procedure in a top-down recursive manner, until the entire margin tree is grown.

This algorithm is exact in that it finds the best split at each node in the top-down tree building process. This is because the best greedy split must be among the candidates considered in step 4, since as mentioned above, all classes in a collapsed node must be on the same side of the decision plane. But it is not exact in a global sense, that is, it does not find the best tree among all possible trees.

Note that if approximation (1) is an equality, then the complete linkage tree is itself the greedy margin classifier solution. This follows because $\hat{M} = M_0$ in the above algorithm.

As an example, consider the problem in Figure 2. We cut the complete linkage tree to produce two nodes (5,4,6) and (1,2,3). We compute the achieved margin for this split and also the margin for partitions (1) vs. (2,3,4,5,6), (2) vs. (1,3,4,5,6) etc. We find that the largest margin corresponds to (1) vs. (2,3,4,5,6), and so this becomes the first split in the greedy tree. We then repeat this process on the daughter subtrees: in this case, just (2,3,4,5,6). Thus we consider (2) vs. (3,4,5,6) , (3) vs (2,4,5,6) etc, as well as the complete linkage split (2,3) vs (4,5,6). The largest margin is achieved by the latter, so me make that split and continue the process.

2.2 Example: 14 Cancer Microarray Data

As an example, we consider the microarray cancer data of Ramaswamy et al. (2001): there are 16,063 genes and 198 samples in 14 classes. The authors provide training and test sets of size 144 and 54 respectively.

The margin trees are shown in Figure 3. The length of each (non-terminal) arm corresponds to the margin that is achieved by the classifier at that split. The final classifiers yielded 18, 18 and 19 errors, respectively on the test set. By comparison, the all-pairs support-vector classifier yielded 20 errors and the nearest centroid classifier had 35 errors. Nearest centroid classification (e.g., Tibshirani et al., 2001) computes the standardized mean feature vector in each class, and then assigns a test sample to the class with the closest centroid. Later we do a more comprehensive comparison of all of these methods. We note that the greedy and single linkage margin tree are "stringy", with each partition separating off just one class in most cases. The complete linkage tree is more balanced, producing some potentially useful subgroupings of the cancer classes.

In this example, full enumeration of the partitions at each node would have required computation of 16,382 two class maximum margin classifiers. The exact greedy algorithm required only 485 such classifiers. In general the cost savings can vary, depending on the height M_0 of the initial cut in the complete linkage tree.

Figure 4 displays the margins that were achieved by each method at their collection of splits. We see that the complete method gives larger margins than the other methods. The largest margin achieved is about 49,000, corresponding to the split between class CNS and Collerectal and so on.. This is larger than the margin between Leukemia and the rest at the top of the greedy tree. This shows that the greediness of the exact algorithm can hurt its overall performance in finding large margin splits.

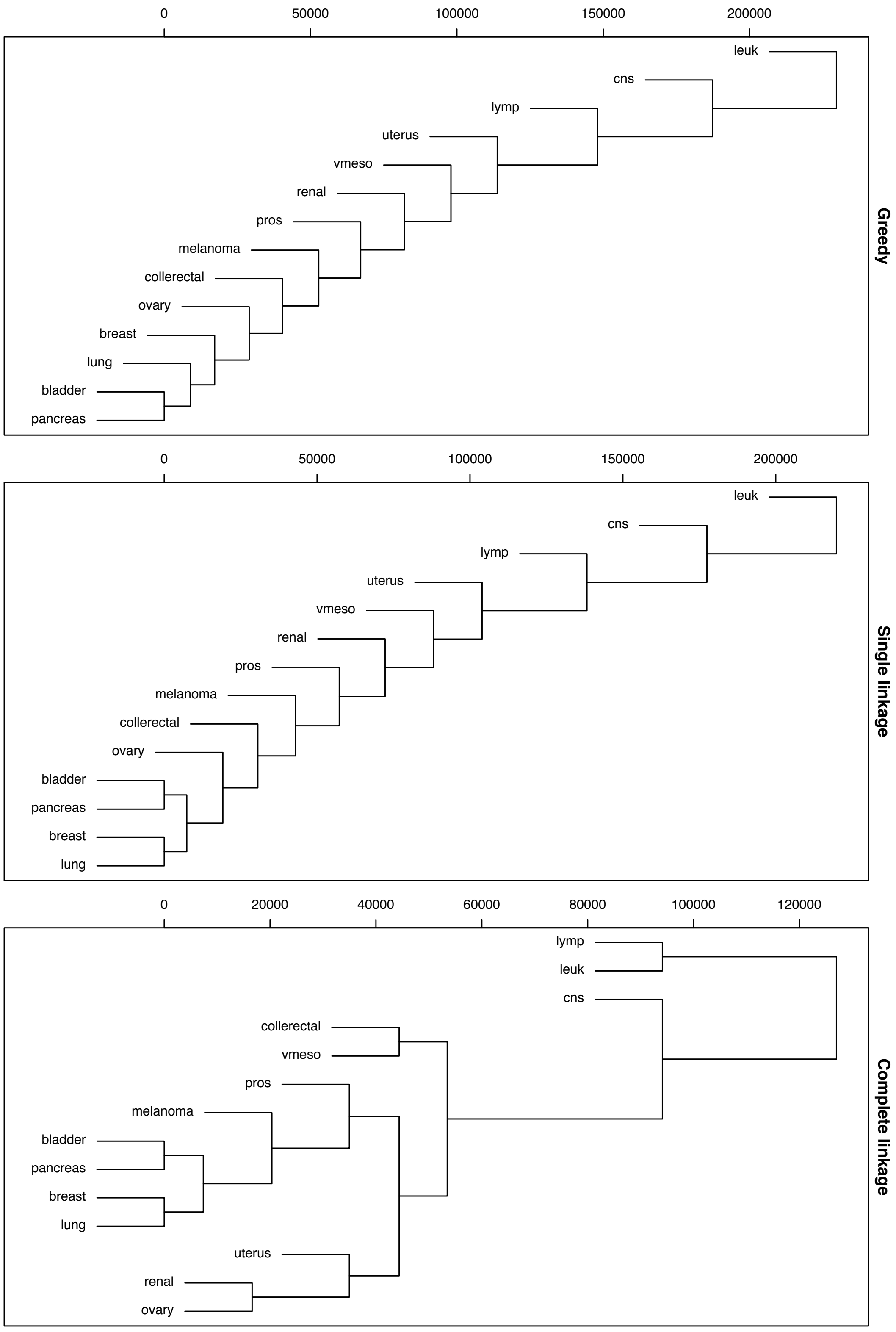


Figure 3: Margin trees for the 14-tumor cancer data of Ramaswamy et al. (2001)

.

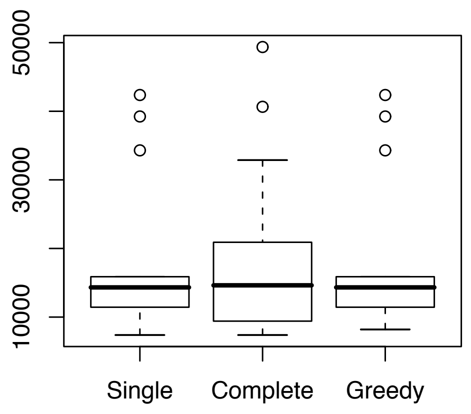


Figure 4: 14 tumor cancer data: margins achieved by each method over the collection of splits. The number of points represented in each boxplot is the number of splits in the corresponding tree.

	SVM(All pairs)	MT(Greedy)	MT(Single)	MT(Complete)
SVM(All pairs)	0	10	11	10
MT(Greedy)	10	0	7	2
MT(Single)	11	7	0	9
MT(Complete)	10	2	9	0

Table 1: Number of disagreements on the test set, for different margin tree-building methods.

Table 1 shows the number of times each classifier disagreed on the test set. The number of disagreements is quite large. However the methods got almost all of the same test cases correct (over 90% overlap), and the disagreements occur almost entirely for test cases in which all methods got the prediction wrong.

Figure 5 shows the test errors at each node of the complete linkage tree, for the 14 tumor data set.

3. Application to Other Cancer Microarray Data Sets

We applied the methods described earlier to the seven microarray cancer data sets shown in Table 2. In each case we randomly sampled 2/3rds of the data to form a training set, and the balance of the data became the test set. The sampling was done in a stratified way to retain balance of the class sizes. This entire process was repeated 50 times, and the mean and standard errors of the test set misclassification rates are shown in Table 3. The nearest centroid method is as described in

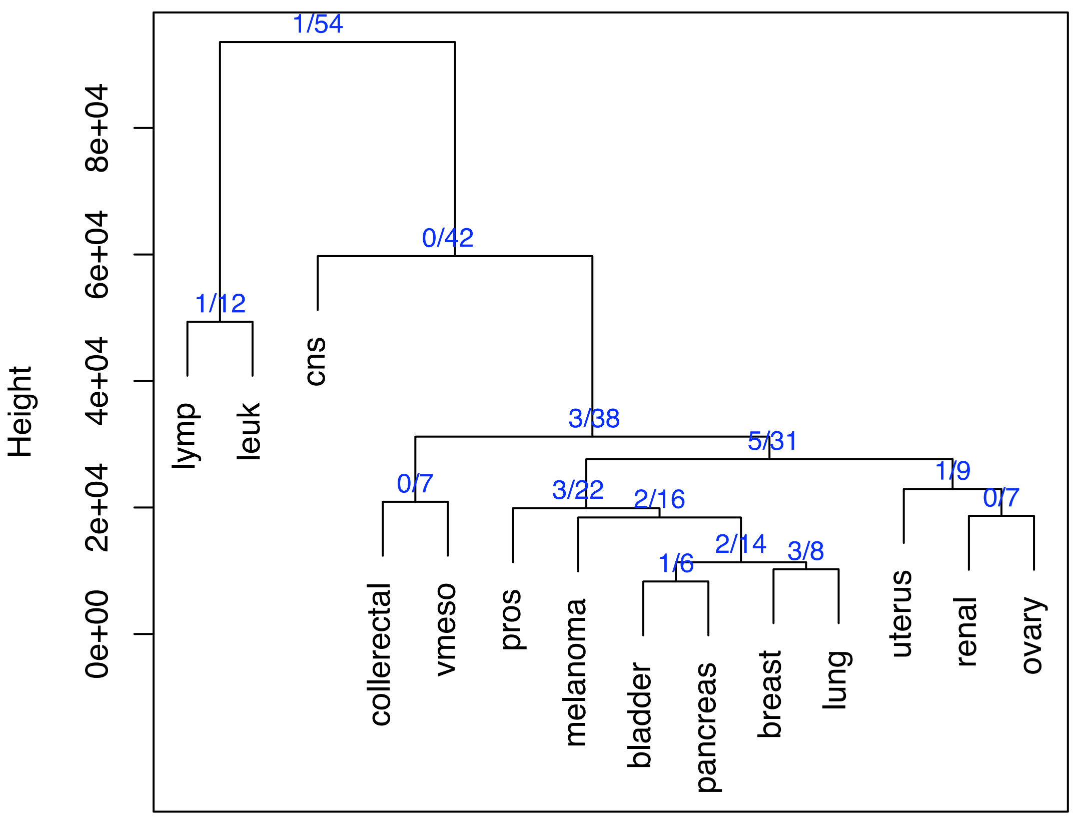


Figure 5: Test errors for the 14 tumor data set using the complete linkage approach. Error rates at each decision junction is shown: notice that the errors tend to increase farther down the tree.

Tibshirani et al. (2001) and uses no shrinkage for feature selection: we discuss feature selection in Section 4. We see that for problems involving more than 4 or 5 classes, the one-versus-one support vector classifier and the margin tree methods sometimes offer an advantage over nearest centroids. The margin tree methods are all very similar to each other and the one-versus-one support vector classifier.

Name	# Classes	# Samples	# Features	Source
Brain	5	22	5597	Pomeroy et al. (2002)
Lymphoma	3	62	4026	Alizadeh et al. (2000)
Small round blue cell tumors	4	63	2308	Khan et al. (2001)
Stanford	14	261	4718	Munagala et al. (2004)
9 tumors	9	60	5726	Staunton et al. (2001)
11 tumors	11	174	12,533	Su et al. (2001)
14 tumors	14	198	16063	Ramaswamy et al. (2001)

Table 2: Summary of data sets for comparative study

.

Data set	Nearest centroids	SVM (OVO)	MT(Single)	MT(Complete)	MT(Greedy)
Brain	0.236(0.026)	0.207(0.022)	0.229(0.018)	0.229(0.021)	0.221(0.017)
Lymphoma	0.010(0.006)	0.000(0.000)	0.000(0.000)	0.000(0.000)	0.000(0.000)
SRBCT	0.065(0.020)	0.011(0.011)	0.014(0.014)	0.014(0.014)	0.014(0.014)
Stanford	0.075(0.006)	0.063(0.006)	0.070(0.005)	0.079(0.007)	0.072(0.005)
9 tumors	0.478(0.009)	0.507(0.014)	0.526(0.013)	0.522(0.014)	0.545(0.014)
11 tumors	0.139(0.006)	0.110(0.005)	0.106(0.005)	0.106(0.005)	0.110(0.005)
14 tumors	0.493(0.007)	0.345(0.006)	0.318(0.007)	0.322(0.007)	0.315(0.007)

Table 3: Mean test error rates (standard errors) over 50 simulations, from various cancer microarray data sets. SVM (OVO) is the support vector machine, using the one-versus-one approach; each pairwise classifier uses a large value for the cost parameter, to yield the maximal margin classifier; MT are the margin tree methods, with different tree-building strategies.

4. Feature Selection

The classifiers at each junction of the margin tree each use all of the features (genes). For interpretability it would be clearly beneficial to reduce the set of genes to a smaller set, if one can improve, or at least not significantly worsen, its accuracy. How one does this depends on the goal.

The investigator probably wants to know which genes have the largest contribution in each classifier. For this purpose, we rank each gene by the absolute value of its coefficient $\hat{\beta}_j$. Then to form a reduced classifier, we simply set to zero the first n_k coefficients at split k in the margin tree. We call this "hard-thresholding".

How do we choose n_k? It is not all clear that n_k should be the same for each tree split. For example we might be able to use fewer genes near the top of the tree, where the margins between the classes is largest.

Our strategy is as follows. We compute reduced classifiers at each tree split, for a range of values of n_k, and for each, the proportion of the full margin achieved by the classifier. Then we use a common value α for the margin proportion throughout the tree. This strategy allows the classifiers at different parts of the tree to use different number of genes. In real applications, we use tenfold cross-validation to estimate the best value for α.

Figure 6 shows the result of applying hard thresholding to the 14-class cancer data. The plot shows the test error as the margin proportion α is varied. The average number of genes at each of

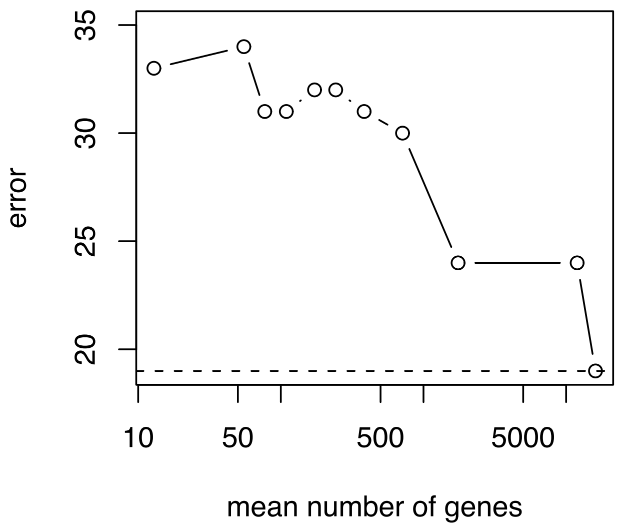


Figure 6: 14 tumor data set: Test errors for reduced numbers of genes.

the 13 tree junctions is shown along the horizontal axis. We see that average number of genes can be reduced from about $16,000$ to about $2,000$ without too much loss of accuracy. But beyond that, the test error increases.

Figure 7 shows a more successful application of the feature selection procedure. The figure shows the result for one training/test split of the 11 class data (12,533 genes) described earlier. With no feature selection the margin tree (left panel) achieves $4/61$ errors, the same as the one-versus one support vector machine. Hard-thresholding (middle panel) also yields 4 errors, with an average of just 167 genes per split. The margin proportion is shown at the top of the plot. The right panel shows the number of genes used as a function of the height of the split in the tree, for margin proportion 0.6.

The feature selection procedure described above is simple and computationally fast. Note that having identified a set of features to be removed, we simply set their coefficients $\hat{\beta}_i$ to zero. For reasons of speed and interpretability, we do not recompute the maximum margin classifier in the subspace of the remaining features (we do however recompute the classifier cutpoint, equal to mid-point between the classes). How much do we lose in this approximation? For the top and bottom splits in the tree of Figure 7, Figure 8 shows the margins achieved by the maximum margin classifier (black points) and the approximation (blue points) as the numbers of genes is reduced. The approximation gives margins remarkably close to the optimal margin until the number of genes drop below 100. Also shown in the figure are the margins achieved by recursive feature elimination (RFE) (Guyon et al., 2002). This is a full backward stepwise procedure, in which successive coefficients are dropped and the optimal margin classifier for the remaining features is recomputed. We see that RFE offers only a small advantage, when the number of genes becomes quite small.

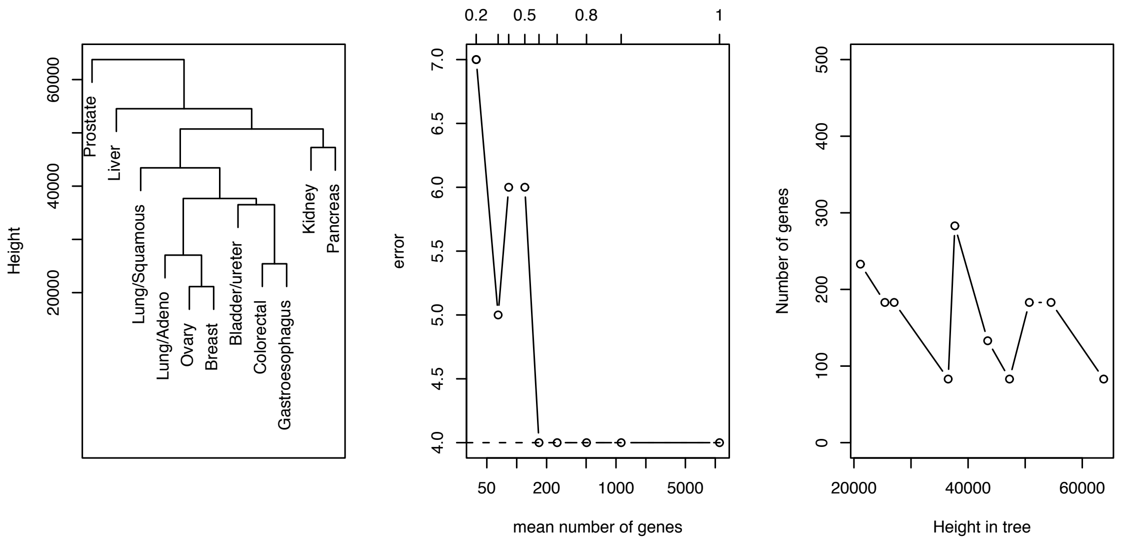


Figure 7: Results for 11 tumor data set. The left panel shows the margin tree using complete linkage; the test errors from hard-thresholding are shown in the middle, with the margin proportion α indicated along the top of the plot; for the tree using $\alpha = 0.6$, the right panel shows the resulting number of genes at each split in the tree, as a function of the height of that split.

4.1 Results on Real Data Sets

Table 4 shows the results of applying the margin tree classifier (complete linkage) with feature selection, on the data sets described earlier. Tenfold cross-validation was used to choose the margin fraction parameter α, and both CV error and test set error are reported in the table. Also shown are results for nearest shrunken centroids (Tibshirani et al., 2001), using cross-validation to choose the shrinkage parameter. This method starts with centroids for each class, and then shrinks them towards the overall centroid by soft-thresholding. We see that (a) hard thresholding generally improves upon the error rate of the full margin tree; (b) margin trees outperform nearest shrunken centroids on the whole, but not in every case. In some cases, the number of genes used has dropped substantially; to get smaller number of genes one could look more closely at the cross-validation curve, to check how quickly it was rising.

If two genes are correlated and both contribute too the classifier, they might both remain in the model, under the above scheme. One the other hand, if there is a set of many highly correlated genes that contribute, their coefficients will be diluted and they might all be removed.

Hence it might be desirable to to select among the genes in a more aggressive fashion. There are a number of approaches one might try here, for example the recursive feature elimination of Guyon et al. (2002), mentioned above. One could also try the L1-norm support-vector machine (see, for example, Zhu et al., 2003), but this is also quite slow to compute. Another approach would be to apply the lasso (Tibshirani, 1997). All of these methods would be worth trying; however they also

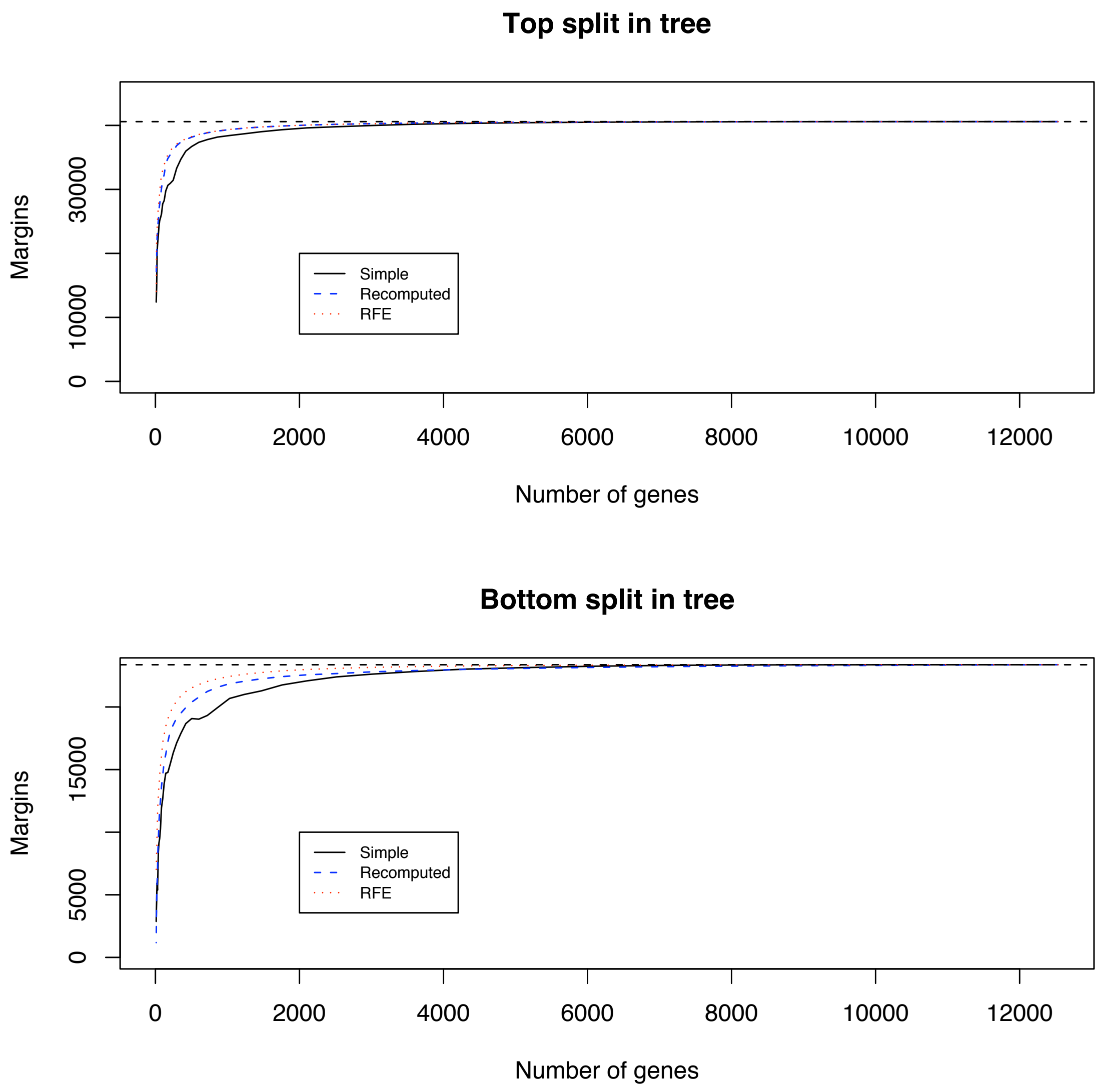


Figure 8: Results for the 11 tumor data set: margins achieved by the maximum margin classifier using simple hard thresholding without recomputing the weights (black points), with recomputation (blue points) and recursive feature elimination (red points). Top panel refers to the top split in the margin tree; bottom panel refers to the bottom split.

suffer from interpretability issues. In particular, the best classifier with say 50 genes might have only a few genes in common with the best classifier with 100 genes. The hard thresholding method described above does not suffer from this drawback. It gives a single ranked list of weights for all genes, for the classifier at each node of the tree.

	Nearest shrunken centroids			Margin tree with selection		
	CV errors	Test errors	# genes used	CV errors	Test errors	# genes/split
Brain	0.192(0.024)	0.276(0.04)	260(119.3)	0.264(0.011)	0.176(0.02)	483.3(194.3)
Lymphoma	0.022(0.002)	0.005(0.005)	3463.4(126.4)	0.000(0.000)	0.01(0.006)	408.5(145.0)
SRBCT	0.00(0.00)	0.014(0.009)	44.7(6.9)	0.004(0.002)	0.010(0.007)	5.08(8.6)
Stanford	0.064(0.005)	0.080(0.012)	4420.1(254.6)	0.064(0.006)	0.076(0.007)	1518(532.4)
9 tumors	0.381(0.008)	0.400(0.016)	1163.4(527.2)	0.503(0.019)	0.476(0.022)	2678.1(847.1)
11 tumors	0.125(0.004)	0.156(0.009)	4857.8(1685.9)	0.111(0.008)	0.110(0.012)	2720.5(1523.9)
14 tumors	0.397(0.007)	0.400(0.018)	3928.5(1392.8)	0.327(0.01)	0.311(0.015)	7563.8(1990.1)

Table 4: CV and test error rates for nearest shrunken centroids and margin trees with feature selection by simple hard thresholding. The rightmost column reports the average number of genes used at each split in the tree.

5. Discussion

The margin-tree method proposed here seems well suited to high-dimensional problems with more than two classes. It has prediction accuracy competitive with multiclass support vector machines and nearest centroid methods, and provides a hierarchical grouping of the classes.

All of the classifiers considered here use a linear kernel, that is, they use the original input features. The construction of margin tree could also be done using other kernels, using the support vector machine framework. The greedy algorithm and linkage algorithms will work without change. However in the $p > N$ case considered in this paper, a linear SVM can separate the data, so the utility of a non-linear kernel is not clear. And importantly, the ability to select features would be lost with a non-linear SVM.

We have restricted attention to the case $p > N$ in which the classes are separable by a hyperplane. When $p < N$ and the classes may not be separable, our approach can be modified to work in principle but may not perform well in practice. The nodes of the clustering tree will be impure, that is contain observations from more than one class. Hence a larger tree—one with more leaves than there are classes—might be needed to effectively classify the observations.

In addition to the papers on the multiclass support vector classifier mentioned earlier, there is other work related to our paper. The decision tree methods of Breiman et al. (1984) ("CART") and Quinlan (1993) use top-down splitting to form a binary tree, but use other criteria (different from the margin) for splitting. With $p \gg N$, splits on individual predictors can get unwieldy and exhibit high variance. The use of linear combination splits is closer to our approach, but again it is not designed for large numbers of predictors. It does not produce a partition of the classes but rather operates on individual observations.

Closely related to CART's linear combination splits is the FACT approach of Loh and Vanichsetakul (1988) and the followup work of Kim and Loh (2001). These use Fisher's linear discriminant function to make multi-way splits of each node of the tree. While linear discriminants might perform similarly to the support vector classifier, the latter has the maximum margin property which we have exploited in this paper.

Probably the closest paper to our work is that of Vural and Dy (2004), who use a top-down binary tree approach. They use K-means clustering of the class means to divide the points in two groups at each node, before applying a support vector classifier. Bennett and Blue (1997) investigate decision trees with support vector classifiers at the node, but do not discuss adaptive construction of the tree topology. Park and Hastie (2005) propose hierarchical classification methods using nearest centroid classifiers at each node. They use clustering methods to find the topology of the tree, and their paper has some ideas in common with this one. In fact, the mixture model used for each merged node gives a decision boundary that is similar to the support vector classifier. However the maximum margin classifier used here seems more natural and the overall performance of the margin tree is better.

Acknowledgments

We would like the thank the referees for helpful comments that led to improvements in this manuscript. Tibshirani was partially supported by National Science Foundation Grant DMS-9971405 and National Institutes of Health Contract N01-HV-28183. Trevor Hastie was partially supported by grant DMS-0505676 from the National Science Foundation, and grant 2R01 CA 72028-07 from the National Institutes of Health.

References

A. Alizadeh, M. Eisen, R. E. Davis, C. Ma, I. Lossos, A. Rosenwal, J. Boldrick, H. Sabet, T. Tran, X. Yu, Pwellm J., G. Marti, T. Moore, J. Hudsom, L. Lu, D. Lewis, R. Tibshirani, G. Sherlock, W. Chan, T. Greiner, D. Weisenburger, K. Armitage, R. Levy, W. Wilson, M. Greve, J. Byrd, D. Botstein, P. Brown, and L. Staudt. Identification of molecularly and clinically distinct substypes of diffuse large b cell lymphoma by gene expression profiling. *Nature*, 403:503–511, 2000.

K. Bennett and J. Blue. A support vector machine approach to decision trees. Technical report, Rensselaer Polytechnic Institute, Troy, NY, 1997. R.P.I Math Report No. 97-100.

L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, pages 389–422, 2002.

J. Khan, J. S. Wei, M. Ringnér, L. H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C. Peterson, and P. S. Meltzer. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, 7:673–679, 2001.

H. Kim and W.Y. Loh. Classification trees with unbiased multiway splits. *Journal of the American Statistical Association*, 96:589–604, 2001.

Y. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data. *Journal of the Amer. Statist. Assoc.*, 99:67–81, 2004.

W.Y. Loh and N. Vanichsetakul. Tree structured classification via generalized discriminant analysis. *Journal of the American Statistical Association*, 83:715–728, 1988.

K. Munagala, R. Tibshirani, and P. Brown. Cancer characterization and feature set extraction by discriminative margin clustering. *BMC Bioinformatics*, 5:5–21, 2004.

M. Y. Park and T. Hastie. Hierarchical classification using shrunken centroids. Technical report, Stanford University, 2005.

S. L. Pomeroy, P Tamayo, M. Gaasenbeek, L. M. Sturla, M. Angelo, M. E. McLaughlin, J. Y. Kim, L. C. Goumnerova, P. M. Black, C. Lau, J. C. Allen, D. Zagzag, J. M. Olson, T. Curran, C. Wetmore, J. A. Biegel, T. Poggio, S. Mukherjee, R. Rifkin, A. Califano, G. Stolovitzky, D. N. Louis, J. P. Mesirov, E. S. Lander, and T. R. Golub. Prediction of central nervous system embryonal tumour outcome based on gene expression. *Nature*, 5:436–42, 2002.

R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, 1993.

S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. Mesirov, T. Poggio, W. Gerald, M. Loda, E. Lander, and T. Golub. Multiclass cancer diagnosis using tumor gene expression signature. *PNAS*, 98:15149–15154, 2001.

S. Rosset, J. Zhu, and T. Hastie. Margin maximizing loss functions. In *Advances in Neural Information Processing Systems, (NIPS*2005)*, 2005.

A. Statnikov, C.F. Aliferis, I. Tsamardinos, D. Hardin, and S. Levy. A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics*, pages 631–43, 2004.

J.E. Staunton, D.K. Slonim, H.A. Coller, P. Tamayo, M.J. Angelo, U. Park, J. Scherf, J.K. Lee, W.O. Reinhold, and J.N. Weinstein. Chemosensitivity prediction by transcriptional profiling. *Proc. Natl Acad. Sci. USA*, 98:10787–10792, 2001.

A.I. Su, J.B. Welsh, L.M. Sapinoso, S.G. Kern, P. Dimitrov, H. Lapp, P.G. Schultz, S.M. Powell, C.A. Moskaluk, H.F. Frierson, Jr, and G.M. Hampton. Molecular classification of human carcinomas by use of gene expression signatures. *Cancer Research*, 61:7388–7393, 2001.

R. Tibshirani. The lasso method for variable selection in the cox model. *Statistics in Medicine*, 16: 385–395, 1997.

R. Tibshirani, T. Hastie, B. Narasimhan, and G Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proc. Natl. Acad. Sci.*, 99:6567–6572, 2001.

V. Vural and J. G. Dy. A hierarchical method for multi-class support vector machines. 2004. International Conference on Machine Learning; Proceeding Series; Vol. 69.

J. Weston and C. Watkins. Multi-class support vector machines. In M. Verleysen, editor, *Proceedings of ESANN99*. D. Facto Press, Brussels, 1999.

J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. L1 norm support vector machines. Technical report, Stanford University, 2003.

Journal of Machine Learning Research 8 (2007) 653-692 Submitted 9/05; Revised 9/06; Published 3/07

Relational Dependency Networks

Jennifer Neville NEVILLE@CS.PURDUE.EDU
Departments of Computer Science and Statistics
Purdue University
West Lafayette, IN 47907-2107, USA

David Jensen JENSEN@CS.UMASS.EDU
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003-4610, USA

Editor: Max Chickering

Abstract

Recent work on graphical models for relational data has demonstrated significant improvements in classification and inference when models represent the dependencies among instances. Despite its use in conventional statistical models, the assumption of instance independence is contradicted by most relational data sets. For example, in citation data there are dependencies among the topics of a paper's references, and in genomic data there are dependencies among the functions of interacting proteins. In this paper, we present relational dependency networks (RDNs), graphical models that are capable of expressing and reasoning with such dependencies in a relational setting. We discuss RDNs in the context of relational Bayes networks and relational Markov networks and outline the relative strengths of RDNs—namely, the ability to represent cyclic dependencies, simple methods for parameter estimation, and efficient structure learning techniques. The strengths of RDNs are due to the use of *pseudolikelihood* learning techniques, which estimate an efficient approximation of the full joint distribution. We present learned RDNs for a number of real-world data sets and evaluate the models in a prediction context, showing that RDNs identify and exploit cyclic relational dependencies to achieve significant performance gains over conventional conditional models. In addition, we use synthetic data to explore model performance under various relational data characteristics, showing that RDN learning and inference techniques are accurate over a wide range of conditions.

Keywords: relational learning, probabilistic relational models, knowledge discovery, graphical models, dependency networks, pseudolikelihood estimation

1. Introduction

Many data sets routinely captured by organizations are relational in nature, yet until recently most machine learning research focused on "flattened" propositional data. Instances in propositional data record the characteristics of homogeneous and statistically independent objects; instances in relational data record the characteristics of heterogeneous objects and the relations among those objects. Examples of relational data include citation graphs, the World Wide Web, genomic structures, fraud detection data, epidemiology data, and data on interrelated people, places, and events extracted from text documents.

The presence of *autocorrelation* provides a strong motivation for using relational techniques for learning and inference. Autocorrelation is a statistical dependency between the values of the same variable on related entities and is a nearly ubiquitous characteristic of relational data sets (Jensen and Neville, 2002). For example, hyperlinked web pages are more likely to share the same topic than randomly selected pages. More formally, we define relational autocorrelation with respect to an attributed graph $G = (V, E)$, where each node $v \in V$ represents an object and each edge $e \in E$ represents a binary relation. Autocorrelation is measured for a set of instance pairs P_R related through paths of length l in a set of edges E_R: $P_R = \{(v_i, v_j) : e_{ik_1}, e_{k_1k_2}, ..., e_{k_lj} \in E_R\}$, where $E_R = \{e_{ij}\} \subseteq E$. It is the correlation between the values of a variable X on the instance pairs $(v_i.x, v_j.x)$ such that $(v_i, v_j) \in P_R$. Recent analyses of relational data sets have reported autocorrelation in the following variables:

- Topics of hyperlinked web pages (Chakrabarti et al., 1998; Taskar et al., 2002),
- Industry categorization of corporations that share board members (Neville and Jensen, 2000),
- Fraud status of cellular customers who call common numbers (Fawcett and Provost, 1997; Cortes et al., 2001),
- Topics of coreferent scientific papers (Taskar et al., 2001; Neville and Jensen, 2003),
- Functions of colocated proteins in a cell (Neville and Jensen, 2002),
- Box-office receipts of movies made by the same studio (Jensen and Neville, 2002),
- Industry categorization of corporations that co-occur in news stories (Bernstein et al., 2003),
- Tuberculosis infection among people in close contact (Getoor et al., 2001), and
- Product/service adoption among customers in close communication (Domingos and Richardson, 2001; Hill et al., 2006).

When relational data exhibit autocorrelation there is a unique opportunity to improve model performance because inferences about one object can inform inferences about related objects. Indeed, recent work in relational domains has shown that *collective inference* over an entire data set results in more accurate predictions than conditional inference for each instance independently (e.g., Chakrabarti et al., 1998; Neville and Jensen, 2000; Lu and Getoor, 2003), and that the gains over conditional models increase as autocorrelation increases (Jensen et al., 2004).

Joint relational models are able to exploit autocorrelation by estimating a joint probability distribution over an entire relational data set and collectively inferring the labels of related instances. Recent research has produced several novel types of graphical models for estimating joint probability distributions for relational data that consist of non-independent and heterogeneous instances (e.g., Getoor et al., 2001; Taskar et al., 2002). We will refer to these models as *probabilistic relational models* (PRMs).[1] PRMs extend traditional graphical models such as Bayesian networks to relational

1. Several previous papers (e.g., Friedman et al., 1999; Getoor et al., 2001) use the term *probabilistic relational model* to refer to a specific model that is now often called a *relational Bayesian network* [Koller, personal communication]. In this paper, we use PRM in its more recent and general sense.

domains, removing the assumption of independent and identically distributed instances that underlies conventional learning techniques.[2] PRMs have been successfully evaluated in several domains, including the World Wide Web, genomic data, and scientific literature.

Directed PRMs, such as relational Bayes networks[3] (RBNs) (Getoor et al., 2001), can model autocorrelation dependencies if they are structured in a manner that respects the acyclicity constraint of the model. While domain knowledge can sometimes be used to structure the autocorrelation dependencies in an acyclic manner, often an acyclic ordering is unknown or does not exist. For example, in genetic pedigree analysis there is autocorrelation among the genes of relatives (Lauritzen and Sheehan, 2003). In this domain, the casual relationship is from ancestor to descendent so we can use the temporal parent-child relationship to structure the dependencies in an acyclic manner (i.e., parents' genes will never be influenced by the genes of their children). However, given a set of hyperlinked web pages, there is little information to use to determine the causal direction of the dependency between their topics. In this case, we can only represent the autocorrelation between two web pages as an undirected correlation. The acyclicity constraint of directed PRMs precludes the learning of arbitrary autocorrelation dependencies and thus severely limits the applicability of these models in relational domains.[4]

Undirected PRMs, such as relational Markov networks (RMNs) (Taskar et al., 2002), can represent and reason with arbitrary forms of autocorrelation. However, research on these models has focused primarily on parameter estimation and inference procedures. Current implementations of RMNs do not select features—model structure must be pre-specified by the user. While, in principle, it is possible for RMN techniques to learn cyclic autocorrelation dependencies, inefficient parameter estimation makes this difficult in practice. Because parameter estimation requires multiple rounds of inference over the entire data set, it is impractical to incorporate it as a subcomponent of feature selection. Recent work on conditional random fields for sequence analysis includes a feature selection algorithm (McCallum, 2003) that could be extended for RMNs. However, the algorithm abandons estimation of the full joint distribution and uses pseudolikelihood estimation, which makes the approach tractable but removes some of the advantages of reasoning with the full joint distribution.

In this paper, we outline relational dependency networks (RDNs),[5] an extension of dependency networks (Heckerman et al., 2000) for relational data. RDNs can represent and reason with the cyclic dependencies required to express and exploit autocorrelation during collective inference. In this regard, they share certain advantages of RMNs and other undirected models of relational data (Chakrabarti et al., 1998; Domingos and Richardson, 2001; Richardson and Domingos, 2006). To our knowledge, RDNs are the first PRM capable of *learning* cyclic autocorrelation dependencies. RDNs also offer a relatively simple method for structure learning and parameter estimation, which results in models that are easier to understand and interpret. In this regard, they share certain advantages of RBNs and other directed models (Sanghai et al., 2003; Heckerman et al., 2004).

2. Another class of joint models extend conventional logic programming models to support probabilistic reasoning in first-order logic environments (Kersting and Raedt, 2002; Richardson and Domingos, 2006). We refer to these models as *probabilistic logic models* (PLMs). See Section 5.2 for more detail.
3. We use the term *relational Bayesian network* to refer to Bayesian networks that have been upgraded to model relational databases. The term has also been used by Jaeger (1997) to refer to Bayesian networks where the nodes correspond to relations and their values represent possible interpretations of those relations in a specific domain.
4. The limitation is due to the PRM modeling approach (see Section 3.1), which ties parameters across items of the same type and can produce cycles in the rolled out inference graph. This issue is discussed in more detail in Section 5.1.
5. This paper continues our previous work on RDNs (Neville and Jensen, 2004).

The primary distinction between RDNs and other existing PRMs is that RDNs are an approximate model. RDNs approximate the full joint distribution and thus are not guaranteed to specify a consistent probability distribution. The quality of the approximation will be determined by the data available for learning—if the models are learned from large data sets, and combined with Monte Carlo inference techniques, the approximation should be sufficiently accurate.

We start by reviewing the details of dependency networks for propositional data. Then we describe the general characteristics of PRMs and outline the specifics of RDN learning and inference procedures. We evaluate RDN learning and inference on synthetic data sets, showing that RDN learning is accurate for large to moderate-sized data sets and that RDN inference is comparable, or superior, to RMN inference over a range of data conditions. In addition, we evaluate RDNs on five real-world data sets, presenting learned RDNs for subjective evaluation. Of particular note, all the real-world data sets exhibit multiple autocorrelation dependencies that were automatically discovered by the RDN learning algorithm. We evaluate the learned models in a prediction context, where only a single attribute is unobserved, and show that the models outperform conventional conditional models on all five tasks. Finally, we review related work and conclude with a discussion of future directions.

2. Dependency Networks

Graphical models represent a joint distribution over a set of variables. The primary distinction between Bayesian networks, Markov networks, and dependency networks (DNs) is that dependency networks are an approximate representation. DNs approximate the joint distribution with a set of conditional probability distributions (CPDs) that are learned independently. This approach to learning results in significant efficiency gains over exact models. However, because the CPDs are learned independently, DNs are not guaranteed to specify a *consistent*[6] joint distribution, where each CPD can be derived from the joint distribution using the rules of probability. This limits the applicability of exact inference techniques. In addition, the correlational DN representation precludes DNs from being used to infer causal relationships. Nevertheless, DNs can encode predictive relationships (i.e., dependence and independence) and Gibbs sampling inference techniques (e.g., Neal, 1993) can be used to recover a full joint distribution, regardless of the consistency of the local CPDs. We begin by reviewing traditional graphical models and then outline the details of dependency networks in this context.

Consider the set of variables $\mathbf{X} = (X_1, ..., X_n)$ over which we would like to model the joint distribution $p(\mathbf{x}) = p(x_1, ..., x_n)$. We use upper case letters to refer to random variables and lower case letters to refer to an assignment of values to the variables.

A Bayesian network for $\mathbf{X}$ uses a directed acyclic graph $G = (V, E)$ and a set of conditional probability distributions P to represent the joint distribution over $\mathbf{X}$. Each node $v \in V$ corresponds to an $X_i \in \mathbf{X}$. The edges of the graph encode dependencies among the variables and can be used to infer conditional independence among variables using notions of d-separation. The parents of node X_i, denoted PA_i, are the set of $v_j \in V$ such that $(v_j, v_i) \in E$. The set P contains a conditional probability distribution for each variable given its parents, $p(x_i|pa_i)$. The acyclicity constraint on G ensures that the CPDs in P factor the joint distribution into the formula below. A directed graph is acyclic if there is no directed path that starts and ends at the same variable. More specifically, there

6. In this paper, we use the term *consistent* to refer to the consistency of the individual CPDs (as Heckerman et al., 2000), rather than the asymptotic properties of a statistical estimator.

can be no self-loops from a variable to itself. Given (G,P), the joint probability for a set of values $\mathbf{x}$ is computed with the formula:

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i|pa_i).$$

A Markov network for $\mathbf{X}$ uses an undirected graph $U = (V,E)$ and a set of potential functions Φ to represent the joint distribution over $\mathbf{X}$. Again, each node $v \in V$ corresponds to an $X_i \in \mathbf{X}$ and the edges of the graph encode conditional independence assumptions. However, with undirected graphs, conditional independence can be inferred using simple graph separation. Let $C(U)$ be the set of cliques in the graph U. Then each clique $c \in C(U)$ is associated with a set of variables X_c and a clique potential $\phi_c(x_c)$ which is a non-negative function over the possible values for x_c. Given (U,Φ), the joint probability for a set of values $\mathbf{x}$ is computed with the formula:

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{i=1}^{c} \phi_i(x_{c_i}),$$

where $Z = \sum_X \prod_{i=1}^{c} \phi_i(x_{c_i})$ is a *normalizing constant*, which sums over all possible instantiations of $\mathbf{x}$ to ensure that $p(\mathbf{x})$ is a true probability distribution.

2.1 DN Representation

Dependency networks are an alternative form of graphical model that approximates the full joint distribution with a set of conditional probability distributions that are each learned independently. A DN encodes probabilistic relationships among a set of variables $\mathbf{X}$ in a manner that combines characteristics of both undirected and directed graphical models. Dependencies among variables are represented with a directed graph $G = (V,E)$, where conditional independence is interpreted using graph separation, as with undirected models. However, as with directed models, dependencies are quantified with a set of conditional probability distributions P. Each node $v_i \in V$ corresponds to an $X_i \in \mathbf{X}$ and is associated with a probability distribution conditioned on the other variables, $P(v_i) = p(x_i|\mathbf{x} - \{x_i\})$. The parents of node i are the set of variables that render X_i conditionally independent of the other variables ($p(x_i|pa_i) = p(x_i|\mathbf{x} - \{x_i\})$), and G contains a directed edge from each parent node v_j to each child node v_i ($(v_j, v_i) \in E$ iff $X_j \in pa_i$). The CPDs in P do not necessarily factor the joint distribution so we cannot compute the joint probability for a set of values $\mathbf{x}$ directly. However, given G and P, a joint distribution can be recovered through Gibbs sampling (see Section 3.4 for details). From the joint distribution, we can extract any probabilities of interest.

For example, the DN in Figure 1 models the set of variables: $\mathbf{X} = \{X_1, X_2, X_3, X_4, X_5\}$. Each node is conditionally independent of the other nodes in the graph given its immediate neighbors (e.g., X_1 is conditionally independent of $\{X_2, X_4\}$ given $\{X_3, X_5\}$). Each node contains a CPD, which specifies a probability distribution over its possible values, given the values of its parents.

2.2 DN Learning

Both the structure and parameters of DNs are determined through learning the local CPDs. The DN learning algorithm learns a separate distribution for each variable X_i, conditioned on the other variables in the data (i.e., $\mathbf{X} - \{X_i\}$). Any conditional learner can be used for this task (e.g., logistic regression, decision trees). The CPD is included in the model as $P(v_i)$ and the variables selected by the conditional learner form the parents of X_i (e.g., if $p(x_i|\{\mathbf{x} - x_i\}) = \alpha x_j + \beta x_k$ then $PA_i = \{x_j, x_k\}$).

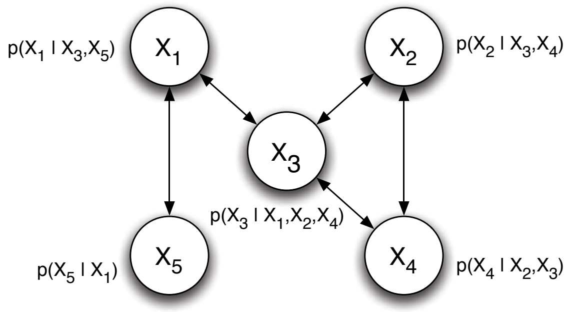


Figure 1: Example dependency network.

The parents are then reflected in the edges of G appropriately. If the conditional learner is not selective (i.e., the algorithm does not select a subset of the features), the DN will be fully connected (i.e., $PA_i = \mathbf{x} - \{x_i\}$). In order to build understandable DNs, it is desirable to use a selective learner that will learn CPDs that use a subset of all available variables.

2.3 DN Inference

Although the DN approach to structure learning is simple and efficient, it can result in an inconsistent network, both structurally and numerically. In other words, there may be no joint distribution from which each of the CPDs can be obtained using the rules of probability. Learning the CPDs independently with a selective conditional learner can result in a network that contains a directed edge from X_i to X_j, but not from X_j to X_i. This is a structural inconsistency—X_i and X_j are dependent but X_j is not represented in the CPD for X_i. In addition, learning the CPDs independently from finite samples may result in numerical inconsistencies in the parameter estimates. If this is the case, the joint distribution derived numerically from the CPDs will not sum to one. However, when a DN is inconsistent, approximate inference techniques can still be used to estimate a full joint distribution and extract probabilities of interest. Gibbs sampling can be used to recover a full joint distribution, regardless of the consistency of the local CPDs, provided that each X_i is discrete and its CPD is positive (Heckerman et al., 2000). In practice, Heckerman et al. (2000) show that DNs are nearly consistent if learned from large data sets because the data serve a coordinating function to ensure some degree of consistency among the CPDs.

3. Relational Dependency Networks

Several characteristics of DNs are particularly desirable for modeling relational data. First, learning a collection of conditional models offers significant efficiency gains over learning a full joint model. This is generally true, but it is even more pertinent to relational settings where the feature space is very large. Second, networks that are easy to interpret and understand aid analysts' assessment of the utility of the relational information. Third, the ability to represent cycles in a network facilitates reasoning with autocorrelation, a common characteristic of relational data. In addition, whereas the need for approximate inference is a disadvantage of DNs for propositional data, due to the complexity of relational model graphs in practice, all PRMs use approximate inference.

Relational dependency networks extend DNs to work with relational data in much the same way that RBNs extend Bayesian networks and RMNs extend Markov networks.[7] These extensions take

7. See Section 5.1 for a more detailed description of RBNs and RMNs.

a graphical model formalism and *upgrade* (Kersting, 2003) it to a first-order logic representation with an entity-relationship model. We start by describing the general characteristics of probabilistic relational models and then discuss the details of RDNs in this context.

3.1 Probabilistic Relational Models

PRMs represent a joint probability distribution over the attributes of a relational data set. When modeling propositional data with a graphical model, there is a single graph G that comprises the model. In contrast, there are three graphs associated with models of relational data: the *data graph* G_D, the *model graph* G_M, and the *inference graph* G_I. These correspond to the *skeleton*, *model*, and *ground graph* as outlined in Heckerman et al. (2004).

First, the relational data set is represented as a typed, attributed data graph $G_D = (V_D, E_D)$. For example, consider the data graph in Figure 2a. The nodes V_D represent objects in the data (e.g., authors, papers) and the edges E_D represent relations among the objects (e.g., author-of, cites).[8] Each node $v_i \in V_D$ and edge $e_j \in E_D$ is associated with a type, $T(v_i) = t_{v_i}$ and $T(e_j) = t_{e_j}$ (e.g., paper, cited-by). Each item[9] type $t \in T$ has a number of associated attributes $\mathbf{X^t} = (X_1^t, ..., X_m^t)$ (e.g., topic, year). Consequently, each object v_i and link e_j is associated with a set of attribute values determined by their type, $\mathbf{X_{v_i}^{t_{v_i}}} = (X_{v_i1}^{t_{v_i}}, ..., X_{v_im}^{t_{v_i}})$ and $\mathbf{X_{e_j}^{t_{e_j}}} = (X_{e_j1}^{t_{e_j}}, ..., X_{e_jm'}^{t_{e_j}})$. A PRM represents a joint distribution over the values of the attributes in the data graph, $\mathbf{x} = \{\mathbf{x_{v_i}^{t_{v_i}}} : v_i \in V \ s.t.\ T(v_i) = t_{v_i}\} \cup \{\mathbf{x_{e_j}^{t_{e_j}}} : e_j \in E \ s.t.\ T(e_j) = t_{e_j}\}$.

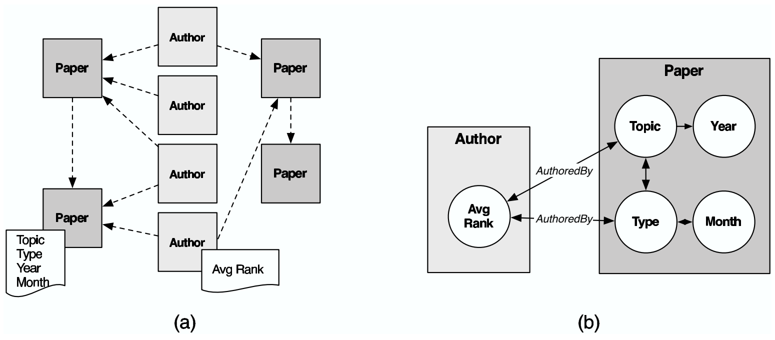


Figure 2: Example (a) data graph and (b) model graph.

Next, the dependencies among attributes are represented in the model graph $G_M = (V_M, E_M)$. Attributes of an item can depend probabilistically on other attributes of the same item, as well as on attributes of other related objects or links in G_D. For example, the topic of a paper may be influenced by attributes of the authors that wrote the paper. The relations in G_D are used to limit the search for possible statistical dependencies, thus they constrain the set of edges that can appear in G_M. However, note that a relationship between two objects in G_D does not necessarily imply a probabilistic dependence between their attributes in G_M.

Instead of defining the dependency structure over attributes of specific objects, PRMs define a generic dependency structure at the level of item types. Each node $v \in V_M$ corresponds to an X_k^t,

8. We use rectangles to represent objects, circles to represent random variables, dashed lines to represent relations, and solid lines to represent probabilistic dependencies.

9. We use the generic term "item" to refer to objects or links.

where $t \in T \wedge X_k^t \in \mathbf{X^t}$. The set of attributes $\mathbf{X_k^t} = (X_{ik}^t : (v_i \in V \vee e_i \in E) \wedge T(i) = t)$ is tied together and modeled as a single variable. This approach of typing items and tying parameters across items of the same type is an essential component of PRM learning. It enables generalization from a *single* instance (i.e., one data graph) by decomposing the data graph into *multiple* examples of each item type (e.g., all paper objects), and building a joint model of dependencies between and among attributes of each type.

As in conventional graphical models, each node is associated with a probability distribution conditioned on the other variables. Parents of X_k^t are either: (1) other attributes associated with items of type t_k (e.g., paper *topic* depends on paper *type*), or (2) attributes associated with items of type t_j where items t_j are related to items t_k in G_D (e.g., paper *topic* depends on author *rank*). For the latter type of dependency, if the relation between t_k and t_j is one-to-many, the parent consists of a set of attribute values (e.g., author ranks). In this situation, current PRMs use aggregation functions to generalize across heterogeneous attributes sets (e.g., one paper may have two authors while another may have five). Aggregation functions are used to either map sets of values into single values, or to combine a set of probability distributions into a single distribution.

Consider the RDN model graph G_M in Figure 2b.[10] It models the data in Figure 2a, which has two object types: paper and author. In G_M, each item type is represented by a plate, and each attribute of each item type is represented as a node. Edges characterize the dependencies among the attributes at the type level. The representation uses a modified plate notation. Dependencies among attributes of the same object are represented by arcs within a rectangle; arcs that cross rectangle boundaries represent dependencies among attributes of related objects, with edge labels indicating the underlying relations. For example, $month_i$ depends on $type_i$, while $avgrank_j$ depends on the $type_k$ and $topic_k$ for all papers k written by author j in G_D.

There is a nearly limitless range of dependencies that could be considered by algorithms for learning PRMs. In propositional data, learners model a fixed set of attributes intrinsic to each object. In contrast, in relational data, learners must decide how much to model (i.e., how much of the relational neighborhood around an item can influence the probability distribution of an item's attributes). For example, a paper's topic may depend of the topics of other papers written by its authors—but what about the topics of the references in those papers or the topics of other papers written by coauthors of those papers? Two common approaches to limiting search in the space of relational dependencies are: (1) exhaustive search of all dependencies within a fixed-distance neighborhood in G_D (e.g., attributes of items up to k links away), or (2) greedy iterative-deepening search, expanding the search in G_D in directions where the dependencies improve the likelihood.

Finally, during inference, a PRM uses a model graph G_M and a data graph G_D to instantiate an inference graph $G_I = (V_I, V_E)$ in a process sometimes called "roll out." The roll out procedure used by PRMs to produce G_I is nearly identical to the process used to instantiate sequence models such as hidden Markov models. G_I represents the probabilistic dependencies among all the variables in a single test set (here G_D is usually different from G'_D used for training). The structure of G_I is determined by both G_D and G_M—each item-attribute pair in G_D gets a separate, local copy of the appropriate CPD from G_M. The relations in G_D determine the way that G_M is rolled out to form G_I. PRMs can produce inference graphs with wide variation in overall and local structure because the structure of G_I is determined by the specific data graph, which typically has non-uniform structure. For example, Figure 3 shows the model from Figure 2b rolled out over the data set in Figure 2a.

10. For clarity, we omit cyclic autocorrelation dependencies in this example. See Section 4.2 for more complex model graphs.

Notice that there are a variable number of authors per paper. This illustrates why current PRMs use aggregation in their CPDs—for example, the CPD for paper-type must be able to deal with a variable number of author ranks.

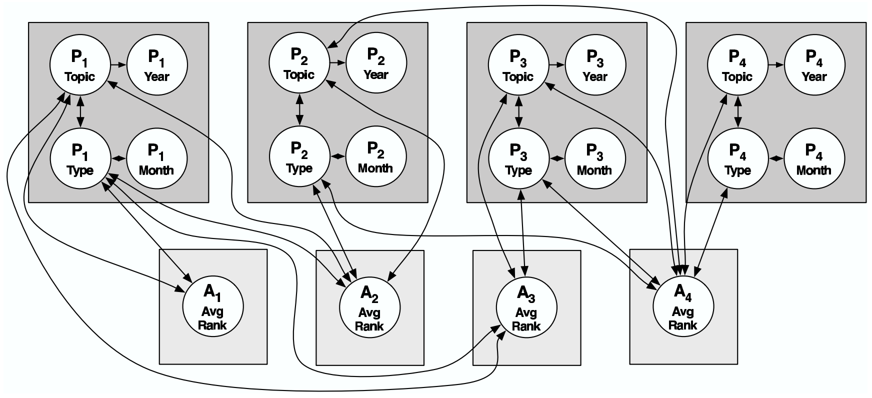


Figure 3: Example inference graph.

3.2 RDN Representation

Relational dependency networks encode probabilistic relationships in a similar manner to DNs, extending the representation to a relational setting. RDNs use a directed model graph G_M with a set of conditional probability distributions P. Each node $v_i \in V_M$ corresponds to an $X_k^t \in \mathbf{X^t}$, $t \in T$ and is associated with a conditional distribution $p(x_k^t \mid pa_{x_k^t})$. Figure 2b illustrates an example RDN model graph for the data graph in Figure 2a. The graphical representation illustrates the qualitative component (G_D) of the RDN—it does not depict the quantitative component (P) of the model, which consists of CPDs that use aggregation functions. Although conditional independence is inferred using an undirected view of the graph, directed edges are useful for representing the set of variables in each CPD. For example, in Figure 2b the CPD for *year* contains *topic* but the CPD for *topic* does not contain *year*. This represents any inconsistencies that result from the RDN learning technique.

A *consistent* RDN specifies a joint probability distribution $p(\mathbf{x})$ over the attribute values of a relational data set from which each CPD $\in P$ can be derived using the rules of probability. There is a direct correspondence between consistent RDNs and relational Markov networks. It is similar to the correspondence between consistent DNs and Markov networks (Heckerman et al., 2000), but the correspondence is defined with respect to the template model graphs G_M and U_M.

Theorem 1 *The set of positive distributions that can be encoded by a consistent RDN (G_M, P) is equal to the set of positive distributions that can be encoded by an RMN (U_M, Φ) provided (1) $G_M = U_M$, and (2) P and Φ use the same aggregation functions.*

Proof Let p be a positive distribution defined by an RMN (U_M, Φ) for G_D. First, we construct a Markov network with tied clique potentials by rolling out the RMN inference graph U_I over the data graph G_D. By Theorem 1 of Heckerman et al. (2000), which uses the Hammersley-Clifford theorem (Besag, 1974), there is a corresponding dependency network that represents the same distribution p as the Markov network U_I. Since the conditional probability distribution for each occurrence of an attribute k of a given type t (i.e., $\forall i\ (v_i \in V_D \vee e_i \in E_D) \wedge T(i) = t\ \ p(x_{ik}^t|\mathbf{x})$) is derived from the Markov network, we know that the resulting CPDs will be identical—the nodes adjacent to each occurrence are equivalent by definition, thus by the global Markov property the derived CPDs will be identical. From this dependency network we can construct a consistent RDN (G_M, P) by first setting $G_M = U_M$. Next, we compute from U_I the CPDs for the attributes of each item type: $p(x_k^t | \mathbf{x} - \{x_k^t\})$ for $t \in T, X_k^t \in \mathbf{X^t}$. To derive the CPDs for P, the CPDs must use the same aggregation functions as the potentials in Φ. Since the adjacencies in the RDN model graph are the same as those in the RMN model graph, and there is a correspondence between the rolled out DN and MN, the distribution encoded by the RDN is p.

Next let p be a positive distribution defined by an RDN (G_M, P) for G_D. First, we construct a dependency network with tied CPDs by rolling out the RDN inference graph G_I over the data graph G_D. Again, by Theorem 1 of Heckerman et al. (2000), there is a corresponding Markov network that represents the same distribution p as the dependency network G_I. Of the valid Markov networks representing p, there will exist a network where the potentials are tied across occurrences of the same clique template (i.e., $\forall c_i \in C \;\; \phi_C(x_C)$). This follows from the first part of the proof, which shows that each RMN with tied clique potentials can be transformed to an RDN with tied CPDs. From this Markov network we can construct an RMN (U_M, Φ) by setting $U_M = G_M$ and grouping the set of clique template potentials in Φ. Since the adjacencies in the RMN model graph are the same as those in the RDN model graph, and since there is a correspondence between the rolled out MN and DN, the distribution encoded by the RMN is p. ■

This proof shows an exact correspondence between *consistent* RDNs and RMNs. We cannot show the same correspondence for general RDNs. However, we will show in Section 3.4 that Gibbs sampling can be used to extract a unique joint distribution, regardless of the consistency of the model.

3.3 RDN Learning

Learning a PRM consists of two tasks: learning the dependency structure among the attributes of each object type, and estimating the parameters of the local probability models for an attribute given its parents. Relatively efficient techniques exist for learning both the structure and parameters of RBNs. However, these techniques exploit the requirement that the CPDs *factor* the full distribution—a requirement that imposes acyclicity constraints on the model and precludes the learning of arbitrary autocorrelation dependencies. On the other hand, it is possible for RMN techniques to learn cyclic autocorrelation dependencies in principle. However, inefficiencies due to calculating the normalizing constant Z in undirected models make this difficult in practice. Calculation of Z requires a summation over all possible states $\mathbf{x}$. When modeling the joint distribution of propositional data, the number of states is exponential in the number of attributes (i.e., $O(2^m)$). When modeling the joint distribution of relational data, the number of states is exponential in the number of attributes and *the number of instances*. If there are N objects, each with m attributes, then the total number of states is $O(2^{Nm})$. For any reasonable-size data set, a single calculation of Z is an enormous computational burden. Feature selection generally requires repeated parameter estimation while measuring the change in likelihood affected by each attribute, which would require recalculation of Z on each iteration.

The RDN learning algorithm uses a more efficient alternative—estimating the set of conditional distributions independently rather than jointly. This approach is based on *pseudolikelihood* techniques (Besag, 1975), which were developed for modeling spatial data sets with similar autocorrelation dependencies. The pseudolikelihood for data graph G_D is computed as a product over the item types t, the attributes of that type X^t, and the items of that type v, e:

$$PL(G_D;\theta) = \prod_{t \in T} \prod_{X_i^t \in \mathbf{X}^t} \prod_{v:T(v)=t} p(x_{vi}^t | pa_{x_{vi}^t};\theta) \prod_{e:T(e)=t} p(x_{ei}^t | pa_{x_{ei}^t};\theta). \tag{1}$$

On the surface, Equation 1 may appear similar to a likelihood that specifies a joint distribution of an RBN. However, the CPDs in the RDN pseudolikelihood are not required to factor the joint distribution of G_D. More specifically, when we consider the variable X_{vi}^t, we condition on the values of the parents $PA_{X_{vi}^t}$ regardless of whether the estimation of CPDs for variables in $PA_{X_{vi}^t}$ was conditioned on X_{vi}^t. The parents of X_{vi}^t may include other variables on the same item (e.g., $X_{vi'}^t$ such that $i' \neq i$), the same variable on related items (e.g., $X_{v'i}^t$ such that $v' \neq v$), or other variables on related items (e.g., $X_{v'i'}^{t'}$ such that $v' \neq v$ and $i' \neq i$).

Pseudolikelihood estimation avoids the complexities of estimating Z and the requirement of acyclicity. Instead of optimizing the log-likelihood of the full joint distribution, we optimize the pseudo-loglikelihood. The contribution for each variable is conditioned on all other attribute values in the data, thus we can maximize the pseudo-loglikelihood for each variable independently:

$$log\, PL(G_D;\theta) = \sum_{t \in T} \sum_{X_i^t \in X^t} \sum_{v:T(v)=t} log\, p(x_{vi}^t | pa_{x_{vi}^t};\theta) + \sum_{e:T(e)=t} log\, p(x_{ei}^t | pa_{x_{ei}^t};\theta).$$

In addition, this approach can make use of existing techniques for learning conditional probability distributions of relational data such as first-order Bayesian classifiers (Flach and Lachiche, 1999), structural logistic regression (Popescul et al., 2003), or ACORA (Perlich and Provost, 2003).

Maximizing the pseudolikelihood function gives the maximum pseudolikelihood estimate (MPLE) of θ. To estimate the parameters we need to solve the following pseudolikelihood equation:

$$\frac{\partial}{\partial\theta} PL(G_D;\theta) = 0. \tag{2}$$

With this approach we lose the asymptotic efficiency properties of maximum likelihood estimators. However, under some general conditions the asymptotic properties of the MPLE can be established. In particular, in the limit as sample size grows, the MPLE will be an unbiased estimate of the true parameter θ_0 and it will be normally distributed. Geman and Graffine (1987) established the first proof of the properties of maximum pseudolikelihood estimators of fully observed data. Gidas (1986) gives an alternative proof and Comets (1992) establishes a more general proof that does not require a finite state space $\mathbf{x}$ or stationarity of the true distribution P_{θ_0}.

Theorem 2 *Assume the following regularity conditions*[11] *are satisfied for an RDN:*

1. *The model is identifiable (i.e., if* $\theta \neq \theta'$*, then* $PL(G_D;\theta) \neq PL(G_D;\theta')$*).*
2. *The distributions* $PL(G_D;\theta)$ *have common support and are differentiable with respect to* θ*.*
3. *The parameter space* Ω *contains an open set* ω *of which the true parameter* θ_0 *is an interior point.*

11. These are the standard regularity conditions (e.g., Casella and Berger, 2002) used to prove asymptotic properties of estimators, which are satisfied in most reasonable problems.

In addition, assume the pseudolikelihood equation (Equation 2) has a unique solution in Ω almost surely as $|G_D| \to \infty$. Then, provided that G_D is of bounded degree, the MPLE $\tilde{\theta}$ converges in probability to the true value θ_0 as $|G_D| \to \infty$.

Proof Provided the size of the RDN does not grow as the size of the data set grows (i.e., $|P|$ remains constant as $|G_D| \to \infty$) and G_D is of bounded degree, then previous proofs apply. We provide the intuition for the proof here and refer the reader to Comets (1992), White (1994), and Lehmann and Casella (1998) for details. Let $\tilde{\theta}$ be the maximum pseudolikelihood estimate that maximizes $PL(G_D;\theta)$. As $|G_D| \to \infty$, the data will consist of all possible data configurations for each CPD $\in P$ (assuming bounded degree structure in G_D). As such, the pseudolikelihood function will converge to its expectation, $PL(G_D;\theta) \to E(PL(G_D;\theta))$. The expectation is maximized by the true parameter θ_0 because the expectation is taken with respect to all possible data configurations. Therefore as $|G_D| \to \infty$, the MPLE converges to the true parameter (i.e., $\tilde{\theta} - \theta_0 \to 0$). ■

The RDN learning algorithm is similar to the DN learning algorithm, except we use a relational probability estimation algorithm to learn the set of conditional models, maximizing pseudolikelihood for each variable separately. The algorithm input consists of: (1) G_D: a relational data graph, (2) R: a conditional relational learner, and (3) $\mathbf{Q^t}$: a set of queries[12] that specify the relational neighborhood considered in R for each type T.

Table 1 outlines the learning algorithm in pseudocode. The algorithm cycles over each attribute of each item type and learns a separate CPD, conditioned on the other values in the training data. We discuss details of the subcomponents (querying and relational learners) in the sections below.

The asymptotic complexity of RDN learning is $O(|\mathbf{X}| \cdot |PA_X| \cdot N)$, where $|\mathbf{X}|$ is the number of CPDs to be estimated, $|PA_X|$ is the number of attributes and N is the number of instances, used to estimate the CPD for X.[13] Quantifying the asymptotic complexity of RBN and RMN learning is difficult due to the use of heuristic search and numerical optimization techniques. RBN learning requires multiple rounds of parameter estimation during the algorithm's heuristic search through the model space, and each round of parameter estimation has the same complexity as RDN learning, thus RBN learning will generally require more time. For RMN learning, there is no closed-form parameter estimation technique. Instead the models are trained using conjugate gradient, where each iteration requires approximate inference over the unrolled Markov network. In general this RMN nested loop of optimization and approximation will require more time to learn than an RBN (Taskar et al., 2002). Therefore, given equivalent search spaces, RMN learning is generally more complex than RBN learning, and RBN learning is generally more complex than RDN learning.

3.3.1 QUERIES

In our implementation, we use queries to specify the relational neighborhoods that will be considered by the conditional learner R. The queries' structures define a typing over instances in the database. Subgraphs are extracted from a larger graph database using the visual query language QGraph (Blau et al., 2001). Queries allow for variation in the number and types of objects and links that form the subgraphs and return collections of all matching subgraphs from the database.

12. Our implementation employs a set of user-specified queries to limit the search space considered during learning. However, a simple depth limit (e.g., ≤ 2 links away in the data graph) can be used to limit the search space as well.
13. This assumes the complexity of the relational learner R is $O(|PA_X| \cdot N)$, which is true for the two relational learners considered in this paper.

```
Learn RDN ($G_D, R, \mathbf{Q^t}$):
$P \leftarrow \emptyset$
For each $t \in T$:
    For each $X_k^t \in \mathbf{X^t}$:
        Use $R$ to learn a CPD for $X_k^t$ given the attributes in the relational
          neighborhood defined by $Q^t$.
        $P \leftarrow P \cup CPD_{X_k^t}$
Use $P$ to form $G_M$.
```

Table 1: RDN learning algorithm.

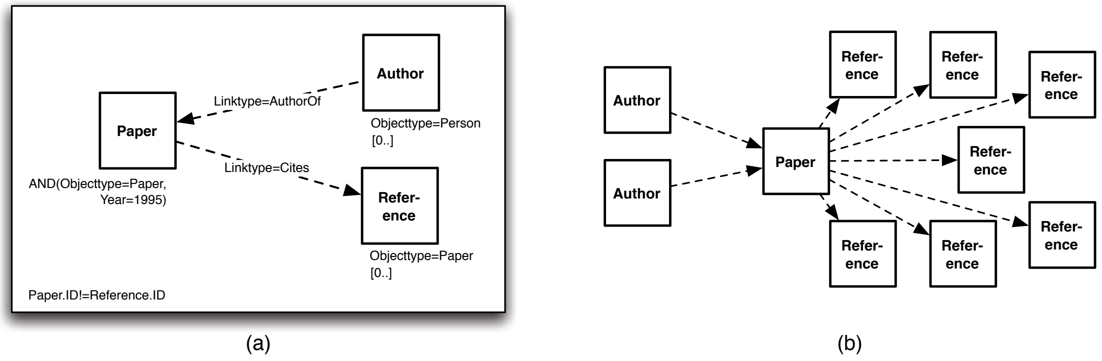


Figure 4: (a) Example QGraph query: Textual annotations specify match conditions on attribute values; numerical annotations (e.g., [0..]) specify constraints on the cardinality of matched objects (e.g., zero or more authors), and (b) matching subgraph.

For example, consider the query in Figure 4a.[14] The query specifies match criteria for a target item (paper) and its local relational neighborhood (authors and references). The example query matches all research papers that were published in 1995 and returns for each paper a subgraph that includes all authors and references associated with the paper. Note the constraint on paper ID in the lower left corner—this ensures that the target paper does not match as a reference in the resulting subgraphs. Figure 4b shows a hypothetical match to this query: a paper with two authors and seven references.

The query defines a typing over the objects of the database (e.g., people that have authored a paper are categorized as *authors*) and specifies the relevant relational context for the target item type in the model. For example, given this query the learner R would model the distribution of a paper's attributes given the attributes of the paper itself and the attributes of its related authors and references. The queries are a means of restricting model search. Instead of setting a simple depth limit on the extent of the search, the analyst has a more flexible means with which to limit the search (e.g., we can consider other papers written by the paper's authors but not other authors of the paper's references).

14. We have modified QGraph's visual representation to conform to our convention of using rectangles to represent objects and dashed lines to represent relations.

3.3.2 CONDITIONAL RELATIONAL LEARNERS

The conditional relational learner R is used for both parameter estimation and structure learning in RDNs. The variables selected by R are reflected in the edges of G_M appropriately. If R selects all of the available attributes, the RDN will be fully connected.

In principle, any conditional relational learner can be used as a subcomponent to learn the individual CPDs provided that it can closely approximate CPDs consistent with the joint distribution. In this paper, we discuss the use of two different conditional models—relational Bayesian classifiers (RBCs) (Neville et al., 2003b) and relational probability trees (RPTs) (Neville et al., 2003a).

Relational Bayesian Classifiers

RBCs extend Bayesian classifiers to a relational setting. RBCs treat heterogeneous relational subgraphs as a homogenous set of attribute multisets. For example, when considering the references of a single paper, the publication dates of those references form multisets of varying size (e.g., $\{1995, 1995, 1996\}$, $\{1975, 1986, 1998, 1998\}$). The RBC assumes each value of a multiset is independently drawn from the same multinomial distribution.[15] This approach is designed to mirror the independence assumption of the naive Bayesian classifier. In addition to the conventional assumption of attribute independence, the RBC also assumes attribute value independence within each multiset.

For a given item type $t \in T$, the query scope specifies the set of item types $\mathbf{T_R}$ that form the relevant relational neighborhood for t. Note that $\mathbf{T_R}$ does not necessarily contain all item types in the database and the query may also dynamically introduce new types in the returned view of the database (e.g., papers $\rightarrow$ papers *and* references). For example, in Figure 4a, $t = paper$ and $\mathbf{T_R} = \{paper, author, reference, authorof, cites\}$. To estimate the CPD for attribute X on items t (e.g., paper topic), the RBC considers all the attributes associated with the types in $\mathbf{T_R}$. RBCs are non-selective models, thus all attributes are included as parents:

$$p(x|pa_x) \propto \prod_{t' \in \mathbf{T_R}} \prod_{X_i^{t'} \in X^{t'}} \prod_{v \in T_R(x)} p(x_{vi}^{t'}|x) \; p(x).$$

Relational Probability Trees

RPTs are selective models that extend classification trees to a relational setting. RPTs also treat heterogeneous relational subgraphs as a set of attribute multisets, but instead of modeling the multisets as independent values drawn from a multinomial, the RPT algorithm uses aggregation functions to map a set of values into a single feature value. For example, when considering the publication dates on references of a research paper, the RPT could construct a feature that tests whether the *average* publication date was after 1995. Figure 5 provides an example RPT learned on citation data.

The RPT algorithm automatically constructs and searches over aggregated relational features to model the distribution of the target variable X on items of type t. The algorithm constructs features from the attributes associated with the types $\mathbf{T_R}$ specified in the query for t. The algorithm considers four classes of aggregation functions to group multiset values: *mode*, *count*, *proportion*, and *degree* (i.e., the number of values in the multiset). For discrete attributes, the algorithm constructs features for all unique values of an attribute. For continuous attributes, the algorithm constructs features for a number of different discretizations, binning the values by frequency (e.g., $year > 1992$). Count, proportion, and degree features consider a number of different thresholds (e.g., $proportion(A) > 10\%$). All experiments reported herein considered 10 thresholds and discretizations per feature.

15. Alternative constructions are possible but prior work (Neville et al., 2003b) has shown this approach achieves superior performance over a wide range of conditions.

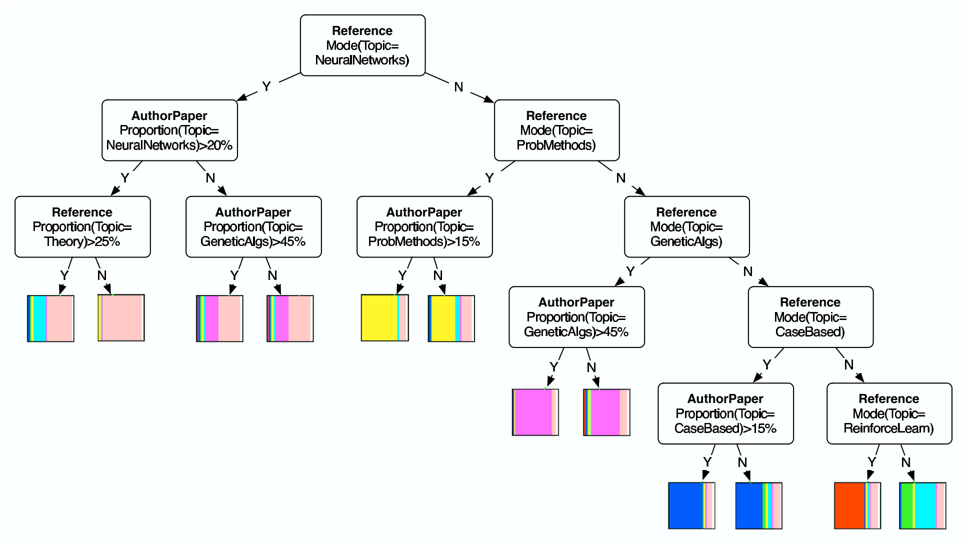

Reference Mode(Topic= NeuralNetworks)
Y
N
AuthorPaper Proportion(Topic= NeuralNetworks)>20%
Reference Mode(Topic= ProbMethods)
Reference Proportion(Topic= Theory)>25%
AuthorPaper Proportion(Topic= GeneticAlgs)>45%
AuthorPaper Proportion(Topic= ProbMethods)>15%
Reference Mode(Topic= GeneticAlgs)
AuthorPaper Proportion(Topic= GeneticAlgs)>45%
Reference Mode(Topic= CaseBased)
AuthorPaper Proportion(Topic= CaseBased)>15%
Reference Mode(Topic= ReinforceLearn)

Figure 5: Example RPT to predict machine-learning paper topic.

The RPT algorithm uses recursive greedy partitioning, splitting on the feature that maximizes the correlation between the feature and the class. Feature scores are calculated using the chi-square statistic and the algorithm uses pre-pruning in the form of a p-value cutoff and a depth cutoff to limit tree size and overfitting. All experiments reported herein used *p-value cutoff*$=0.05/|attributes|$, *depth cutoff*$=7$. Although the objective function does not optimize pseudolikelihood directly, probability estimation trees can be used effectively to approximate CPDs consistent with the underlying joint distribution (Heckerman et al., 2000).

The RPT learning algorithm adjusts for biases towards particular features due to degree disparity and autocorrelation in relational data (Jensen and Neville, 2002, 2003). We have shown that RPTs build significantly smaller trees than other conditional models and achieve equivalent, or better, performance (Neville et al., 2003a). These characteristics of RPTs are crucial for learning understandable RDNs and have a direct impact on inference efficiency because smaller trees limit the size of the final inference graph.

3.4 RDN Inference

The RDN inference graph G_I is potentially much larger than the original data graph. To model the full joint distribution there must be a separate node (and CPD) for each attribute value in G_D. To construct G_I, the set of template CPDs in P is rolled out over the test-set data graph. Each item-attribute pair gets a separate, local copy of the appropriate CPD. Consequently, the total number of nodes in the inference graph will be $\sum_{v \in V_D} |\mathbf{X}^{\mathbf{T}(\mathbf{v})}| + \sum_{e \in E_D} |\mathbf{X}^{\mathbf{T}(\mathbf{e})}|$. Roll out facilitates generalization across data graphs of varying size—we can learn the CPD templates from one data graph and apply the model to a second data graph with a different number of objects by rolling out more CPD copies. This approach is analogous to other graphical models that tie distributions across the network and roll out copies of model templates (e.g., hidden Markov models, conditional random fields (Lafferty et al., 2001)).

We use Gibbs samplers for inference in RDNs. This refers to a procedure where a random ordering of the variables is selected; each variable is initialized to an arbitrary value; and then each variable is visited (repeatedly) in order, where its value is resampled according to its conditional distribution. Gibbs sampling can be used to extract a unique joint distribution, regardless of the consistency of the model.

Theorem 3 *The procedure of a Gibbs sampler applied to an RDN (G,P), where each X_i is discrete and each local distribution in P is positive, defines a Markov chain with a unique stationary joint distribution $\tilde{\pi}$ for $\mathbf{X}$ that can be reached from any initial state of the chain.*

Proof The proof that Gibbs sampling can be used to estimate the joint distribution of a dependency network (Heckerman et al., 2000) applies to rolled out RDNs as well. We restate the proof here for completeness.

Let $\mathbf{x}^t$ be the sample of $\mathbf{x}$ after the t^{th} iteration of the Gibbs sampler. The sequence $\mathbf{x}^1, \mathbf{x}^2, ...$ can be viewed as samples drawn from a homogeneous Markov chain with transition matrix $\tilde{\mathbf{P}}$, where $\tilde{\mathbf{P}}_{ij} = p(\mathbf{x}^{t+1} = j | \mathbf{x}^t = i)$. The matrix $\tilde{\mathbf{P}}$ is the product $\tilde{\mathbf{P}}^1 \cdot \tilde{\mathbf{P}}^2 \cdot ... \cdot \tilde{\mathbf{P}}^n$, where $\tilde{\mathbf{P}}^k$ is the *local* transition matrix describing the resampling of X^k according to the local distribution of $p(x_k|\mathbf{pa}_k)$. The positivity of the local distributions guarantees the positivity of $\tilde{\mathbf{P}}$. The positivity of $\tilde{\mathbf{P}}$ in turn guarantees that the Markov chain is irreducible and aperiodic. Consequently there exists a unique joint distribution that is stationary with respect to $\tilde{\mathbf{P}}$, and this stationary distribution can be reached from any starting point. ■

This shows that a Gibbs sampling procedure can be used with an RDN to recover samples from a unique stationary distribution $\tilde{\pi}$, but how close will this distribution be to the true distribution π? Small perturbations in the local CPDs could propagate in the Gibbs sampling procedure to produce large deviations in the stationary distribution. Heckerman et al. (2000) provide some initial theoretical analysis that suggests that Markov chains with good convergence properties will be insensitive to deviations in the transition matrix. This implies that when Gibbs sampling is effective (i.e., converges), then $\tilde{\pi}$ will be close to π and the RDN will be a close approximation to the full joint distribution.

Table 2 outlines the inference algorithm. To estimate a joint distribution, we start by rolling out the model G_M onto the target data set G_D and forming the inference graph G_I. The values of all unobserved variables are initialized to values drawn from prior distributions, which we estimate empirically from the training set. Gibbs sampling then iteratively relabels each unobserved variable by drawing from its local conditional distribution, given the current state of the rest of the graph. After a sufficient number of iterations (*burn in*), the values will be drawn from a stationary distribution and we can use the samples to estimate probabilities of interest.

For prediction tasks, we are often interested in the marginal probabilities associated with a single variable X (e.g., paper topic). Although Gibbs sampling may be a relatively inefficient approach to estimating the probability associated with a joint assignment of values of X (e.g., when $|X|$ is large), it is often reasonably fast to use Gibbs sampling to estimate the marginal probabilities for each X.

There are many implementation issues that can improve the estimates obtained from a Gibbs sampling chain, such as length of burn-in and number of samples. For the experiments reported in this paper, we used fixed-length chains of 2000 samples (each iteration re-labels every value sequentially) with burn-in set at 100. Empirical inspection indicated that the majority of chains had converged by 500 samples. Section 4.1 includes convergence graphs for synthetic data experiments.

```
**Infer RDN** $(G_D, G_M, P, iter, burnin)$:

$G_I(V_I, E_I) \leftarrow (\emptyset, \emptyset)$                    \\ *form $G_I$ from $G_D$ and $G_M$*
For each $t \in T$ *in* $G_M$:
    For each $X_k^t \in \mathbf{X^t}$ *in* $G_M$:
        For each $v_i \in V_D$ $s.t.$ $T(v_i) = t$ and $e_i \in E_D$ $s.t.$ $T(e_i) = t$:
            $V_I \leftarrow V_I \cup \{X_{ik}^t\}$
        For each $v_i \in V_D$ $s.t.$ $T(v_i) = t$ and $e_i \in E_D$ $s.t.$ $T(e_i) = t$:
            For each $v_j \in V_D$ $s.t.$ $X_{v_j} \in pa_{X_{ik}^t}$ and each $e_j \in E_D$ $s.t.$ $X_{e_j} \in pa_{X_{ik}^t}$:
                $E_I \leftarrow E_I \cup \{e_{ij}\}$

For each $v \in V_I$:                    \\ *initialize Gibbs sampling*
    Randomly initialize $x_v$ to value drawn from prior distribution $p(x_v)$

$S \leftarrow \emptyset$                    \\ *Gibbs sampling procedure*
Choose a random ordering over $V_I$
For $i \in iter$:
    For each $v \in V_I$, in random order:
        Resample $x'_v$ from $p(x_v | \mathbf{x} - \{x_v\})$
        $x_v \leftarrow x'_v$
        If $i > burnin$:
            $S \leftarrow S \cup \{\mathbf{x}\}$:
Use samples $S$ to estimate probabilities of interest
```

Table 2: RDN inference algorithm.

4. Experiments

The experiments in this section demonstrate the utility of RDNs as a joint model of relational data. First, we use synthetic data to assess the impact of training-set size and autocorrelation on RDN learning and inference, showing that accurate models can be learned with reasonable data set sizes and that the model is robust to varying levels of autocorrelation. In addition, to assess the quality of the RDN approximation for inference, we compare RDNs to RMNs, showing that RDNs achieve equivalent or better performance over a range of data sets. Next, we learn RDNs of five real-world data sets to illustrate the types of domain knowledge that the models discover automatically. In addition, we evaluate RDNs in a prediction context, where only a single attribute is unobserved in the test set, and report significant performance gains compared to two conditional models.

4.1 Synthetic Data Experiments

To explore the effects of training-set size and autocorrelation on RDN learning and inference, we generated homogeneous data graphs with an autocorrelated class label and linkage due to an underlying (hidden) group structure. Each object has four boolean attributes: X_1, X_2, X_3, and X_4. We used the following generative process for a data set with N_O objects and N_G groups:

```
For each object $i$, $1 \leq i \leq N_O$:
    Choose a group $g_i$ uniformly from the range $[1, N_G]$.
For each object $j$, $1 \leq j \leq N_O$:
    For each object $k$, $j < k \leq N_O$:
        Choose whether the two objects are linked from $p(E|G_j = G_k)$, a Bernoulli
          probability conditioned on whether the two objects are in the same group.
For each object $i$, $1 \leq i \leq N_O$:
    Randomly initialize the values of $\mathbf{X} = \{X_1, X_2, X_3, X_4\}$ from a uniform prior dis-
      tribution.
Update the values of $\mathbf{X}$ with 500 iterations of Gibbs sampling using $RDN^*$, a manually
  specified model.[16]
```

The data generation procedure for $\mathbf{X}$ uses a manually specified model where X_1 is autocorrelated (through objects one link away), X_2 depends on X_1, and the other two attribute have no dependencies. To generate data with autocorrelated X_1 values, we used conditional models for $p(X_1|\mathbf{X_1}_R, X_2, X_3, X_4)$. $RPT_{0.5}$ refers to the RPT CPD that is used to generate data with autocorrelation levels of 0.5. $RBC_{0.5}$ refers to the analogous RBC CPD. Appendix A contains detailed specifications of these models. Unless otherwise specified, the experiments use the settings below:

$$\begin{aligned}
N_O &= 250, \\
N_G &= \frac{N_O}{10}, \\
p(E|G_j{=}G_k) &= \{p(E{=}1|G_j{=}G_k) = 0.50;\ p(E{=}1|G_j{\neq}G_k) = \frac{1}{N_O}\}, \\
RDN^* &=: [\,p(X_1|\mathbf{X_1}_R, X_2, X_3, X_4) = p(X_1|\mathbf{X_1}_R, X_2) = RPT_{0.5} \text{ or } RBC_{0.5}; \\
&\qquad p(X_2|X_1) = \{p(X_2{=}1|X_1{=}1) = p(X_2{=}0|X_1{=}0) = 0.75\}; \\
&\qquad p(X_3 = 1) = p(X_4 = 1) = 0.50\,].
\end{aligned}$$

4.1.1 RDN LEARNING

The first set of synthetic experiments examines the effectiveness of the RDN learning algorithm. We learned CPDs for X_1 using the intrinsic attributes of the object (X_2, X_3, X_4) as well as the class label of directly related objects (X_{1R}). We also learned CPDs for each attribute (X_2, X_3, X_4) using the class label (X_1). This mimics the structure of the true model used for data generation (i.e., RDN^*).

We compared two different learned RDNs: RDN_{RBC} uses RBCs for the component learner R; RDN_{RPT} uses RPTs for R. The RPT performs feature selection, which may result in structural inconsistencies in the learned RDN. The RBC does not use feature selection so any deviation from the true model is due to parameter inconsistencies alone. Note that the two models do not consider identical feature spaces so we can only roughly assess the impact of feature selection by comparing RDN_{RBC} and RDN_{RPT} results.

Theoretical analysis indicates that, in the limit, the true parameters will maximize the pseudolikelihood function. This indicates that the pseudolikelihood function, evaluated at the learned

16. We will use a star (i.e., RDN^*) to denote manually-specified RDNs.

parameters, will be no greater than the pseudolikelihood of the true model (on average). To evaluate the quality of the RDN parameter estimates, we calculated the pseudolikelihood of the test-set data using both the true models (RDN^*_{RPT}, RDN^*_{RBC}) and the learned models (RDN_{RPT}, RDN_{RBC}). If the pseudolikelihood given the learned parameters approaches the pseudolikelihood given the true parameters, then we can conclude that parameter estimation is successful. We also measured the standard error of the pseudolikelihood estimate for a single test-set using learned models from 10 different training sets. This illustrates the amount of variance due to parameter estimation.

Figure 6 graphs the pseudo-loglikelihood of learned models as a function of training-set size for three levels of autocorrelation. Training-set size was varied at the levels $\{50, 100, 250, 500, 1000, 5000\}$. We varied $p(X_1|\mathbf{X_1}_R, X_2)$ to generate data with approximate levels of autocorrelation corresponding to $\{0.25, 0.50, 0.75\}$. At each training set size (and autocorrelation level), we generated 10 test sets. For each test set, we generated 10 training sets and learned RDNs. Using each learned model, we measured the pseudolikelihood of the test set (size 250) and averaged the results over the 10 models. We plot the mean pseudolikelihood for both the learned models and the true models. The top row reports experiments with data generated from an RDN^*_{RPT}, where we learned an RDN_{RPT}. The bottom row reports experiments with data generated from an RDN^*_{RBC}, where we learned an RDN_{RBC}.

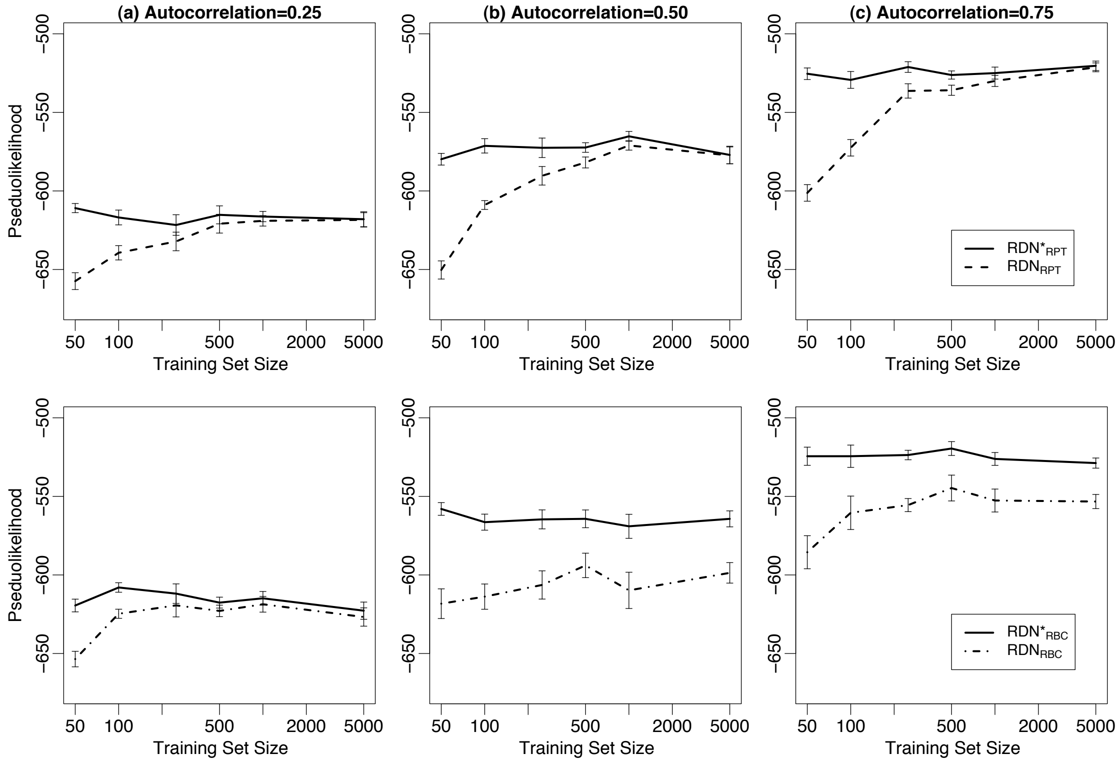


Figure 6: Evaluation of RDN learning.

These experiments show that the learned RDN_{RPT} is a good approximation to the true model by the time training-set size reaches 500, and that RDN learning is robust with respect to varying levels of autocorrelation.

There appears to be little difference between the RDN_{RPT} and RDN_{RBC} when autocorrelation is low, but otherwise the RDN_{RBC} needs significantly more data to estimate the parameters accurately. One possible source of error is variance due to lack of selectivity in the RDN_{RBC}, which necessitates the estimation of a greater number of parameters. However, there is little improvement even when we increase the size of the training sets to 10,000 objects. Furthermore, the discrepancy between the estimated model and the true model is greatest when autocorrelation is moderate. This indicates that the inaccuracies may be due to the naive Bayes independence assumption and its tendency to produce biased probability estimates (Zadrozny and Elkan, 2001).

4.1.2 RDN INFERENCE

The second set of synthetic experiments evaluates the RDN inference procedure in a prediction context, where only a single attribute is unobserved in the test set. We generated data with the RDN^*_{RPT} and RDN^*_{RBC} as described above and learned models for X_1 using the intrinsic attributes of the object (X_2, X_3, X_4) as well as the class label and the attributes of directly related objects $(X_{1R}, X_{2R}, X_{3R}, X_{4R})$. At each autocorrelation level, we generated 10 training sets (size 500) to learn the models. For each training set, we generated 10 test sets (size 250) and used the learned models to infer marginal probabilities for X_1 on the test set instances. To evaluate the predictions, we report area under the ROC curve (AUC).[17] These experiments used the same levels of autocorrelation outlined above.

We compare the performance of four types of models. First, we measure the performance of RPTs and RBCs. These are *conditional models* that reason about each instance independently and do not use the class labels of related instances. Second, we measure the performance of learned RDNs: RDN_{RBC} and RDN_{RPT}. For RDN inference, we used fixed-length Gibbs chains of 2000 samples with burn-in of 100. Third, we measure performance of the learned RDNs while allowing the true labels of related instances to be used during inference. This demonstrates the level of performance possible if the RDNs could infer the true labels of related instances with perfect accuracy. We refer to these as *ceiling models*: RDN^{ceil}_{RBC} and RDN^{ceil}_{RPT}. Fourth, we measure the performance of two RMNs described below.

The first RMN is non-selective. We construct features from all the attributes available to the RDNs, defining clique templates for each pairwise combination of class label value and attribute value. More specifically, the available attributes consist of the intrinsic attributes of objects, and both the class label and attributes of directly related objects. The second RMN, which we refer to as RMN_{Sel}, is a hybrid selective model—clique templates are only specified for the set of attributes selected by the RDN during learning. For both models, we used maximum-a-posteriori parameter estimation to estimate the feature weights, using conjugate gradient with zero-mean Gaussian priors, and a uniform prior variance of 5.[18] For RMN inference, we used loopy belief propagation (Murphy et al., 1999).

We do not compare directly to RBNs because their acyclicity constraint prevents them from representing the autocorrelation dependencies in this domain. Instead, we include the performance of conditional models, which also cannot represent the autocorrelation of X_1. Although RBNs and conditional models cannot represent the autocorrelation directly, they can exploit the autocorrelation indirectly by using the observed attributes of related instances. For example, if there is a

17. Squared-loss results are qualitatively similar to the AUC results reported in Figure 7.
18. We experimented with a range of priors; this parameter setting produced the best empirical results.

correlation between the words on a webpage and its topic, and the topics of hyperlinked pages are autocorrelated, then the models can exploit autocorrelation dependencies by modeling the contents of a webpage's neighboring pages. Recent work has shown that collective models (e.g., RDNs) are a low-variance means of reducing bias through direct modeling of the autocorrelation dependencies (Jensen et al., 2004). Models that exploit autocorrelation dependencies indirectly by modeling the observed attributes of related instances, experience a dramatic increase in variance as the number of observed attributes increases.

During inference we varied the number of known class labels in the test set, measuring performance on the remaining unlabeled instances. This serves to illustrate model performance as the amount of information seeding the inference process increases. We expect performance to be similar when other information seeds the inference process—for example, when some labels can be inferred from intrinsic attributes, or when weak predictions about many related instances serve to constrain the system. Figure 7 graphs AUC results for each model as the proportion of known class labels is varied.

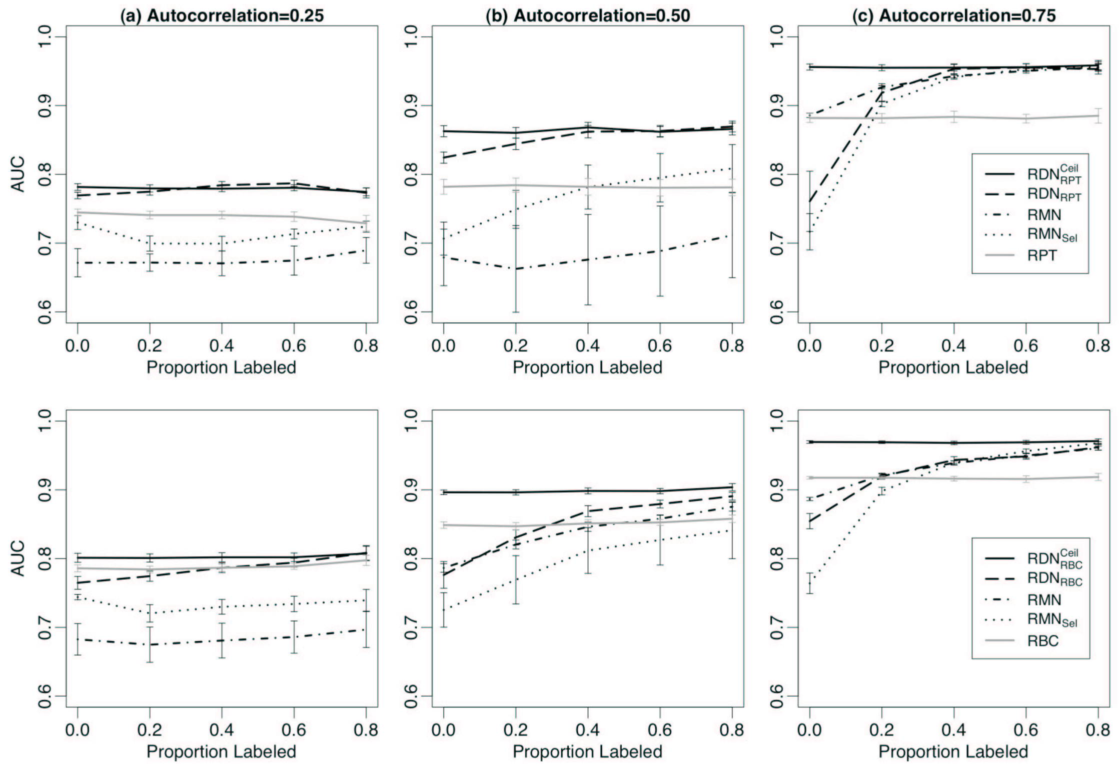


Figure 7: Evaluation of RDN inference.

The data for the first set of experiments (top row) were generated with an RDN^*_{RPT}. In all configurations, RDN_{RPT} performance is equivalent, or better than, RPT performance. This indicates that even modest levels of autocorrelation can be exploited to improve predictions using an RDN_{RPT}. RDN_{RPT} performance is indistinguishable from that of RDN^{ceil}_{RPT} except when autocorrelation is high and there are no labels to seed inference. In this situation, the predictive attribute values (i.e., X_2) are the only information available to constrain the system during inference so the model cannot fully exploit the autocorrelation dependencies. When there is no information to anchor the predictions, there is an identifiability problem—symmetric labelings that are highly autocorrelated, but with opposite values, appear equally likely. In situations where there is little seed information (either attributes or class labels), identifiability problems can increase variance and bias *RDN* performance towards random.

When there is low or moderate autocorrelation, RDN_{RPT} performance is significantly higher than both RMNs. In these situations, poor RMN performance is likely due to a mismatch in feature space with the data generation model—if the RMN features cannot represent the data dependencies that are generated with aggregated features, the inferred probabilities will be biased. When there is high autocorrelation, RDN_{RPT} performance is indistinguishable from *RMN*, except when there are no labels to seed inference—the same situation where RDN_{RPT} fails to meet its ceiling. When autocorrelation is high, the mismatch in feature space is not a problem. In this situation most neighbors share similar attribute values, thus the RMN features are able to accurately capture the data dependencies.

The data for the second set of experiments (bottom row) were generated with an RDN^{*}_{RBC}. The RDN_{RBC} feature space is roughly comparable to the RMN because the RDN_{RBC} uses multinomials to model individual neighbor attribute values. On these data, RDN_{RBC} performance is superior to *RMN* performance only when there is low autocorrelation. RMN_{Sel} uses fewer features than *RMN* and it has superior performance on the data with low autocorrelation, indicating that the RMN learning algorithm may be overfitting the feature weights and producing biased probability estimates. We experimented with a range of priors to limit the impact of weight overfitting, but the effect remained consistent.

RDN_{RBC} performance is superior to *RBC* performance only when there is moderate to high autocorrelation and sufficient seed information. When autocorrelation is low, the RBC is comparable to both the RDN^{ceil}_{RBC} and the RDN_{RBC}. Even when autocorrelation is moderate or high, RBC performance is still relatively high. Since the RBC is low-variance and there are only four attributes in our data sets, it is not surprising that the RBC is able to exploit autocorrelation to improve performance. What is more surprising is that RDN_{RBC} requires substantially more seed information than RDN_{RPT} in order to reach ceiling performance. This indicates that our choice of model should take test-set characteristics (e.g., number of known labels) into consideration.

To investigate Gibbs sampling convergence, we tracked AUC throughout the RDN Gibbs sampling procedure. Figure 8 demonstrates AUC convergence on each inference task described above. We selected a single learned model at random from each task and report convergence from the trials corresponding to five different test sets. AUC improves very quickly, often leveling off within the first 250 iterations. This shows that the approximate inference techniques employed by the RDN may be quite efficient to use in practice. However, when autocorrelation is high, longer chains may be necessary to ensure convergence. There are only two chains that show a substantial increase in performance after 500 iterations and both occur in highly autocorrelated data sets. Also, the RDN_{RBC} chains exhibit significantly more variance than the RDN_{RPT} chains, particularly when autocorrelation is high. This may indicate that the use of longer Gibbs chains, or an approach that averages predictions obtained from multiple random restarts, would improve performance.

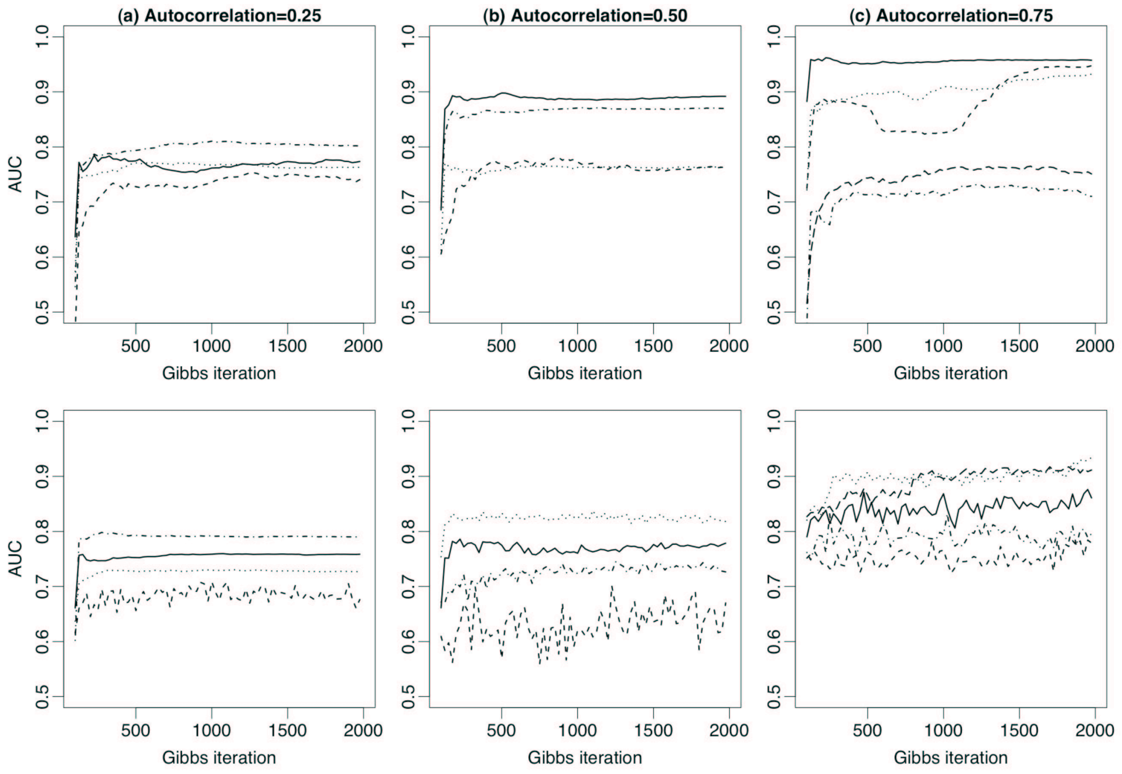


Figure 8: Gibbs convergence rates for five different trials of *RDN*$_{RPT}$ (top row) and *RDN*$_{RBC}$ (bottom row).

4.2 Empirical Data Experiments

We learned RDNs for five real-world relational data sets. Figure 9 depicts the objects and relations in each data set. Section 4.2.1 illustrates the types of domain knowledge that can be learned with the selective *RDN*$_{RPT}$. Section 4.2.2 evaluates both the *RDN*$_{RPT}$ and the *RDN*$_{RBC}$ in a prediction context, where the values of a single attribute are unobserved.

The first data set is drawn from Cora, a database of computer science research papers extracted automatically from the web using machine learning techniques (McCallum et al., 1999). We selected the set of 4,330 machine-learning papers along with associated authors, cited papers, and journals. The resulting collection contains approximately 13,000 objects and 26,000 links. For classification, we sampled the 1669 papers published between 1993 and 1998.

The second data set (Gene) is a relational data set containing information about the yeast genome at the gene and the protein level.[19] The data set contains information about 1,243 genes and 1,734 interactions among their associated proteins.

The third data set is drawn from the Internet Movie Database (IMDb).[20] We collected a sample of 1,382 movies released in the United States between 1996 and 2001, with their associated actors, directors, and studios. In total, this sample contains approximately 42,000 objects and 61,000 links.

19. See http://www.cs.wisc.edu/~dpage/kddcup2001/.
20. See http://www.imdb.com.

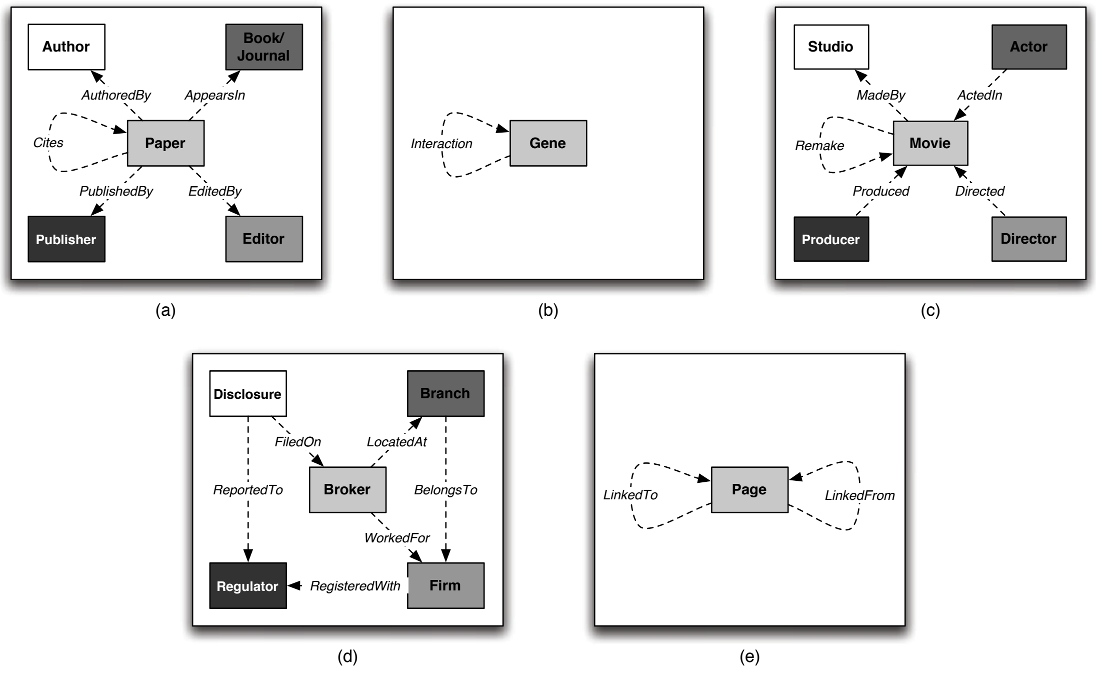


Figure 9: Data schemas for (a) Cora, (b) Gene, (c) IMDb, (d) NASD, and (e) WebkKB.

The fourth data set is from the National Association of Securities Dealers (NASD) (Neville et al., 2005). It is drawn from NASD's Central Registration Depository (CRD©) system, which contains data on approximately 3.4 million securities brokers, 360,000 branches, 25,000 firms, and 550,000 disclosure events. Disclosures record disciplinary information on brokers, including information on civil judicial actions, customer complaints, and termination actions. Our analysis was restricted to small and moderate-size firms with fewer than 15 brokers, each of whom has an approved NASD registration. We selected a set of approximately 10,000 brokers who were active in the years 1997-2001, along with 12,000 associated branches, firms, and disclosures.

The fifth data set was collected by the WebKB Project (Craven et al., 1998). The data comprise 3,877 web pages from four computer science departments. The web pages have been manually labeled with the categories course, faculty, staff, student, research project, or other. The collection contains approximately 8,000 hyperlinks among the pages.

4.2.1 RDN MODELS

The RDNs in Figures 10-14 continue with the RDN representation introduced in Figure 2b. Each item type is represented by a separate plate. An arc from x to y indicates the presence of one or more features of x in the conditional model learned for y. Arcs inside a plate represent dependencies among the attributes of a single object. Arcs crossing plate boundaries represent dependencies among attributes of related objects, with edge labels indicating the underlying relations. When the dependency is on attributes of objects more than a single link away, the arc is labeled with a small rectangle to indicate the intervening related-object type. For example, in Figure 10 paper topic is influenced by the topics of other papers written by the paper's authors, so the arc is labeled with two *AuthoredBy* relations and a small *A* rectangle indicating an *Author* object.

In addition to dependencies among attribute values, relational learners may also learn dependencies between the structure of relations (edges in G_D) and attribute values. *Degree* relationships are represented by a small black circle in the corner of each plate—arcs from this circle indicate a dependency between the number of related objects and an attribute value of an object. For example, in Figure 10 author rank is influenced by the number of paper written by the author.

For each data set, we learned an RDN_{RPT} with queries that included all neighbors up to two links away in the data graph. For example in Cora, when learning an RPT of a paper attribute, we considered the attributes of associated authors and journals, as well as papers related to those objects.

On the Cora data, we learned an RDN for seven attributes. Author rank records ordering in paper authorship (e.g., first author, second author). Paper type records category information (e.g., PhD thesis, technical report); topic records content information (e.g., genetic algorithms, reinforcement learning); year and month record publication dates. Journal name-prefix records the first four title letters (e.g., IEEE, SIAM); book-role records type information (e.g., workshop, conference).

Figure 10 shows the resulting RDN. The RDN learning algorithm selected 12 of the 139 dependencies considered for inclusion in the model. Four of the attributes—author rank, paper topic, paper type, and paper year—exhibit autocorrelation dependencies. In particular, the topic of a paper depends not only on the topics of other papers that it cites, but also on the topics of other papers written by the authors. This model is a good reflection of our domain knowledge about machine learning papers.

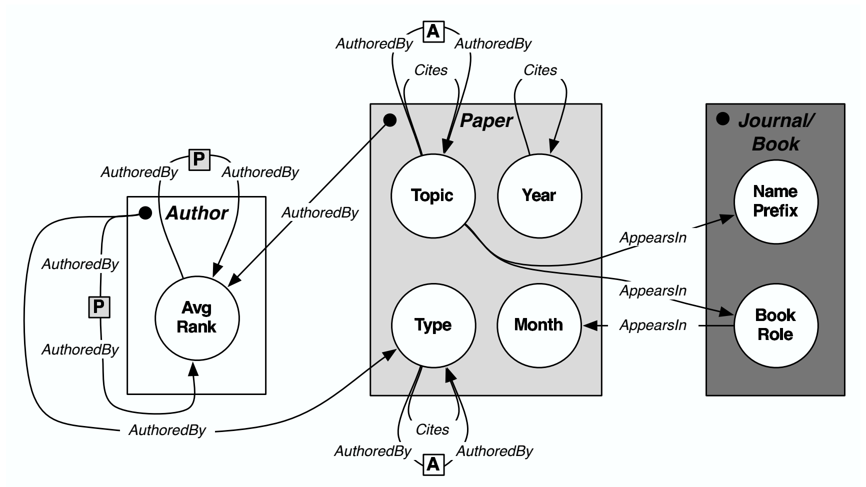


Figure 10: RDN for the Cora data set.

Exploiting these types of autocorrelation dependencies has been shown to significantly improve classification accuracy of RMNs compared to RBNs, which cannot model cyclic dependencies (Taskar et al., 2002). However, to exploit autocorrelation, RMNs must be instantiated with the appropriate clique templates—to date there is no RMN algorithm for *learning* autocorrelation dependencies. RDNs are the first PRM capable of learning cyclic autocorrelation dependencies.

The Gene data contain attributes associated with both objects and links (i.e., interactions). We learned an RDN for seven attributes. Gene function records activities of the proteins encoded by the genes; location records each protein's localization in the cell; phenotype records characteristics of individuals with a mutation in the gene/protein; class records the protein type (e.g., transcription factor, protease); essential records whether the gene is crucial to an organism's survival. Interaction expression records the correlation between gene expression patterns for pairs of interacting genes; type records interaction characteristics (e.g., genetic, physical).

Figure 11 shows the resulting RDN. The RDN learning algorithm selected 19 of the 77 dependencies considered for inclusion in the model. In these data, all the gene attributes exhibit autocorrelation—this is strong evidence that there are regularities among the genes whose proteins interact in the cell.

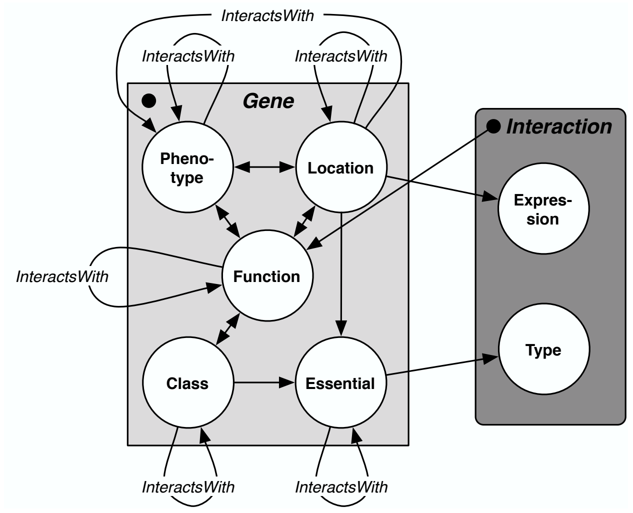


Figure 11: RDN for the Gene data set.

On the IMDb data, we learned an RDN for ten discrete attributes. First-movie-year records the date of the first movie made by a director or studio; has-award records whether a director or actor has won an Academy award; in-US records whether a studio is located in the US; receipts records whether a movie made more than $2 million in the opening weekend box office; genre records a movie's type (e.g., drama, comedy); hsx-rating records an actor's value on the Hollywood Stock Exchange (www.hsx.com); birth-year and gender record demographic information.

Figure 12 shows the resulting RDN. The RDN learning algorithm selected 29 of the 170 dependencies considered for inclusion in the model. Again we see that four of the attributes exhibit autocorrelation. Movie receipts and genre each exhibit a number of autocorrelation dependencies (through actors, directors, and studios), which illustrates the group structure of the Hollywood movie industry.

On the NASD data, we learned an RDN for eleven attributes. Firm size records the number of employed stockbrokers each year; layoffs records the number of terminations each year. On-watchlist records whether a firm or broker is under heightened supervision. Broker is-problem and problem-in-past record whether a broker is, or has been, involved in serious misconduct; has-business records whether a broker owns a business on the side. Disclosure type and year record category (e.g., customer complaint) and date information regarding disciplinary events (filed on brokers). Region and area record location information about branches.

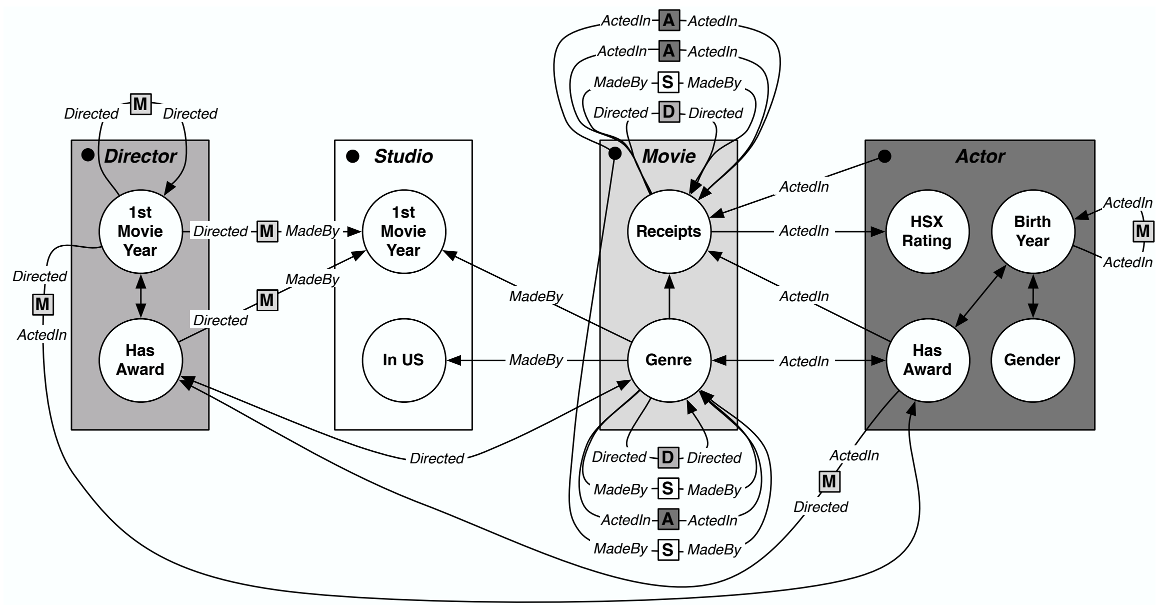


Figure 12: RDN for the IMDb data set.

Figure 13 shows the resulting RDN. The RDN learning algorithm selected 32 of the 160 dependencies considered for inclusion in the model. Again we see that four of the attributes exhibit autocorrelation. Subjective inspection by NASD analysts indicates that the RDN had automatically uncovered statistical relationships that confirm the intuition of domain experts. These include temporal autocorrelation of risk (past problems are indicators of future problems) and relational autocorrelation of risk among brokers at the same branch—indeed, fraud and malfeasance are usually social phenomena, communicated and encouraged by the presence of other individuals who also wish to commit fraud (Cortes et al., 2001). Importantly, this evaluation was facilitated by the interpretability of the RDN—experts are more likely to trust, and make regular use of, models they can understand.

On the WebKB data, we learned an RDN for four attributes. School records page location (e.g., Cornell, Washington); label records page type (e.g., student, course); URL-server records the first portion of the server name following *www* (e.g., cs, engr); URL-text records the name of the first directory in the URL path (e.g., UTCS, department). Figure 14 shows the resulting RDN. The RDN learning algorithm selected 9 of the 52 dependencies considered for inclusion in the model. All of the attributes exhibit autocorrelation in the WebKB data.

4.2.2 PREDICTION

We evaluated RDNs on prediction tasks in order to assess (1) whether autocorrelation dependencies among instances can be used to improve model accuracy, and (2) whether the RDNs, using Gibbs sampling, can effectively infer labels for a network of instances. To do this, we compared the same four classes of models used in Section 4.1: conditional models, RDNs, ceiling RDNs, and RMNs.

We used the following prediction tasks: movie receipts for IMDb, paper topic for Cora, page label for WebKB, gene location for Gene, and broker is-problem for NASD. For each data set, we used queries that included all neighbors up to two links away in the data graph to learn RDN_{RBC} and

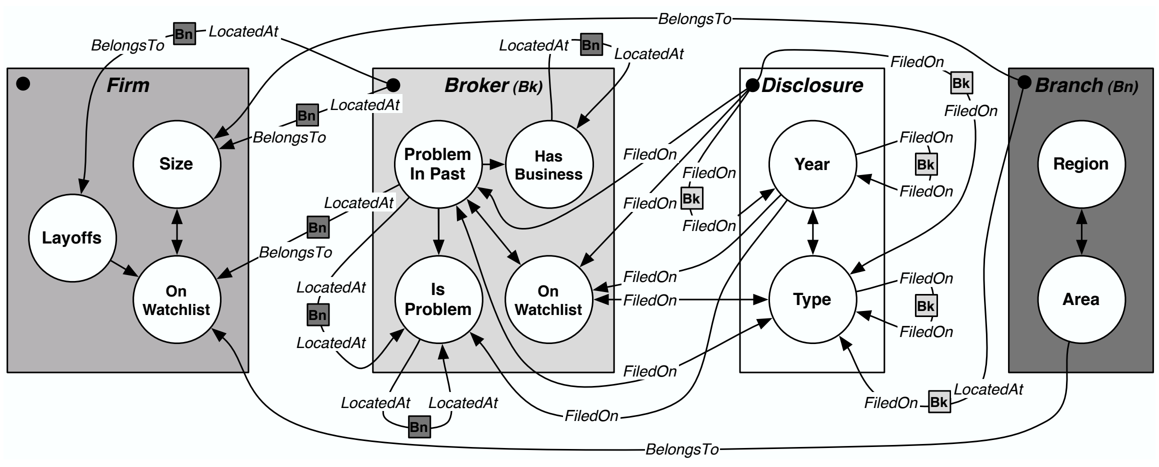


Figure 13: RDN for the NASD data set (1999).

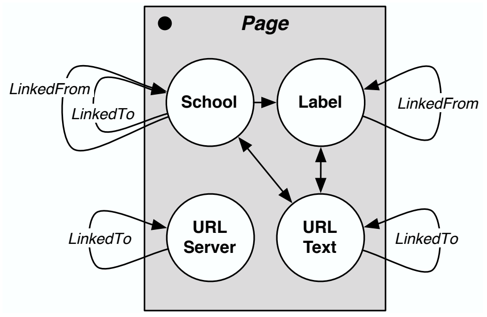


Figure 14: RDN for the WebKB data set.

RDN_{RPT} models. Recall that the RDN_{RBC} uses the entire set of attributes in the resulting subgraphs and the RDN_{RPT} performs feature selection over the attribute set.

Figure 15 shows AUC results for the first three model types on the five prediction tasks. (We discuss RMN results below.) Figure 15a graphs the results of the RDN_{RPT}, compared to the *RPT* conditional model. Figure 15b graphs the results of the RDN_{RBC}, compared to the *RBC* conditional model.

The graphs show AUC for the most prevalent class, averaged over a number of training/test splits. For Cora, IMDb, and NASD, we used temporal sampling where we learned models on one year of data and applied the models to the subsequent year. There were four temporal samples for IMDb and NASD, and five for Cora. For WebKB we used cross-validation by department, learning on three departments and testing on pages from the fourth, held-out department. For Gene there was no clear sampling choice, so we used ten-fold cross validation on random samples of genes. When there were links between the test and training sets, the class labels of the training set were made available to the RDNs and RMNs for use during inference. We used two-tailed, paired t-tests to assess the significance of the AUC results obtained from the trials. The t-tests compare the RDN results to the conditional and ceiling models, with a null hypothesis of no difference in the AUC.

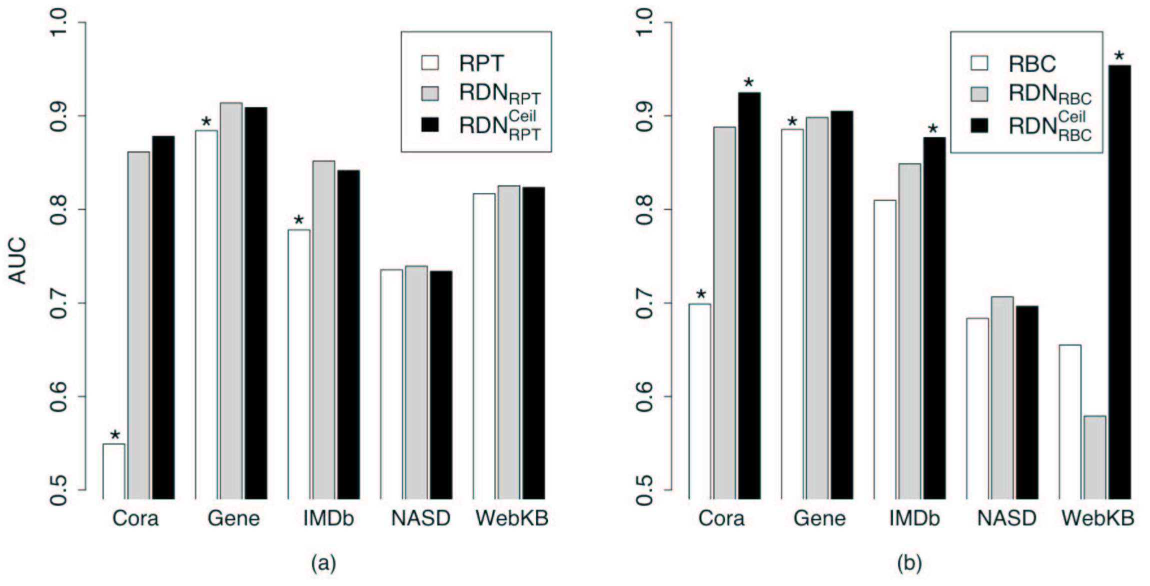


Figure 15: AUC results for (a) RDN_{RPT} and RPT, and (b) RDN_{RBC} and RBC. Asterisks denote model performance that is significantly different ($p < 0.10$) from RDN_{RPT} and RDN_{RBC}.

When using the RPT as the conditional learner (Figure 15a), RDN performance is superior to RPT performance on all tasks. The difference is statistically significant for three of the five tasks. This indicates that autocorrelation is both present in the data and identified by the RDNs. As mentioned previously, the RPT can sometimes use the observed attributes of related items to effectively reason with autocorrelation dependencies. However, in some cases the observed attributes contain little information about the class labels of related instances. This is the case for Cora—RPT performance is close to random because no other attributes influence paper topic (see Figure 10). On all tasks, the RDNs achieve comparable performance to the ceiling models. This indicates that the RDN achieved the same level of performance as if it had access to the true labels of related objects. We note, however, that the ceiling model only represents a probabilistic ceiling—the RDN may perform better if an incorrect prediction for one object improves inferences about related objects. Indeed, on a number of the data sets, RDN performance is slightly higher than that of the ceiling model.

Similarly, when using the RBC as the conditional learner (Figure 15b), the performance of RDNs is superior to the RBCs on all but one task and statistically significant for two of the tasks. Notably, on the WebKB data RDN performance is worse than that of the RBC. However, the ceiling performance is significantly higher than RBC. This indicates that autocorrelation dependencies are identified by the RDN but the model is unable to exploit those dependencies during Gibbs sampling. This effect is due to the amount of information available to seed the inference process. There is sparse information in the attributes other than page label, and because the departments are nearly disjoint, there are few labeled instances before inference. This leaves the RDN with little information to anchor its predictions, which results in marginal predictions closer to random. Similar behavior appeared in the synthetic data experiments, indicating that the RDN_{RBC} may need more information to seed the inference process.

	Num *RDN* **attributes**	**Num selected** *RDN* **attributes**	**Num** *RMN* **features**	**Num** RMN_{Sel} **features**
Cora	22	2.0	1029	98.0
Gene	19	3.1	3640	606.0
IMDb	15	5.0	234	90.5
NASD	16	5.7	526	341.5
WebKB	34	5.0	2478	270.0

Table 3: Number of attributes/features used by RDNs and RMNs.

The RDN_{RBC} achieves comparable performance to the ceiling models on only two of the five tasks. This may be another indication that RDNs combined with a non-selective conditional learner (e.g., RBCs) will experience increased variance during the Gibbs sampling process, and thus they may need more seed information during inference to achieve the near-ceiling performance. We should note that although the RDN_{RBC} does not significantly outperform the RDN_{RPT} on any of the tasks, the RDN^{Ceil}_{RBC} is significantly higher than RDN^{Ceil}_{RPT} for Cora and IMDb. This indicates that, when there is enough seed information, the RDN_{RBC} may achieve significant performance gains over the RDN_{RPT}.

Due to time and memory constraints, we were unable to learn RMNs for all but two of the real-data experiments. Table 3 reports information for each data set—the number of attributes available to the RDNs, the average number of attributes selected by the RDNs, and the number of features constructed by the *RMN* and the RMN_{Sel}. Recall that the *RMN* constructs its features from all the attributes available to the RDN and the RMN_{Sel} constructs its features from the attributes selected by the RDN. Due to the size of the feature spaces considered by the non-selective *RMN*, we were unable learn models for any of the data sets. We were able to successfully learn an RMN_{Sel} for the Cora and IMDb data sets. The average AUC of the RMN_{Sel} was 74.4% for Cora and 60.9% for IMDb. This is far below the RDN results reported in Figure 15. Note that previous RMN results (Taskar et al., 2002) report accuracy of the most likely labeling for the entire data set. In contrast, we are evaluating AUC of the marginal probabilities for each instance (inferred jointly). This may indicate that RMN inference will produce biased (marginal) probability estimates when run in a "loopy" relational network, due to overfitting the clique weights. When skewed weights are applied to collectively infer the labels throughout the test set, the inference process may converge to extreme labelings (e.g., all positive labels in some regions of the graph, all negative labels in other regions), which would bias probability estimates for many of the instances.

5. Related Work

There are three types of statistical relational models relevant to RDNs: probabilistic relational models, probabilistic logic models, and collective inference models. We discuss each of these below.

5.1 Probabilistic Relational Models

Probabilistic relational models are one class of models for density estimation in relational data sets. Examples of PRMs include relational Bayesian networks and relational Markov networks.

As outlined in Section 3.1, learning and inference in PRMs involve a *data graph* G_D, a *model graph* G_M, and an *inference graph* G_I. All PRMs model data that can be represented as a graph (i.e., G_D). PRMs use different approximation techniques for inference in G_I (e.g., Gibbs sampling, loopy belief propagation), but they all use a similar process for rolling out an inference graph G_I. Consequently, PRMs differ primarily with respect to the representation of the model graph G_M and how that model is learned.

An RBN for $\mathbf{X}$ uses a directed model graph $G_M = (V_M, E_M)$ and a set of conditional probability distributions P to represent a joint distribution over $\mathbf{X}$. Each node $v \in V_M$ corresponds to an $X_k^t \in \mathbf{X}$. The set P contains a conditional probability distribution for each variable given its parents, $p(x_k^t | pa_{x_k^t})$. Given (G_M, P), the joint probability for a set of values $\mathbf{x}$ is computed as a product over the item types T, the attributes of that type X^t, and the items of that type v, e:

$$p(\mathbf{x}) = \prod_{t \in T} \prod_{X_i^t \in X^t} \prod_{v: T(v)=t} p(x_{vi}^t | pa_{x_{vi}^t}) \prod_{e: T(e)=t} p(x_{ei}^t | pa_{x_{ei}^t}).$$

The RBN learning algorithm (Getoor et al., 2001) for the most part uses standard Bayesian network techniques for parameter estimation and structure learning. One notable exception is that the learning algorithm must check for "legal" structures that are guaranteed to be acyclic when rolled out for inference on arbitrary data graphs. In addition, instead of exhaustive search of the space of relational dependencies, the structure learning algorithm uses greedy iterative-deepening, expanding the search in directions where the dependencies improve the likelihood.

The strengths of RBNs include understandable knowledge representations and efficient learning techniques. For relational tasks, with a huge space of possible dependencies, *selective* models are easier to interpret and understand than *non-selective* models. Closed-form parameter estimation techniques allow for efficient structure learning (i.e., feature selection). Also, because reasoning with relational models requires more space and computational resources, efficient learning techniques make relational modeling both practical and feasible.

The directed acyclic graph structure is the underlying reason for the efficiency of RBN learning. As mentioned in Section 1, the acyclicity requirement precludes the learning of arbitrary autocorrelation dependencies and limits the applicability of these models in relational domains. The PRM representation, which ties parameters across items of the same type, makes it difficult for RBNs to represent autocorrelation dependencies. Because autocorrelation is a dependency among the value of the same variable on linked items of the same type, the CPDs that represent autocorrelation will produce cycles in the rolled out inference graph. For example, when modeling the autocorrelation among the topics of two hyperlinked web pages P_1 and P_2, the CPD for the topic of P_1 will have the topic of P_2 as a parent and vice versa. In general, unless there is causal knowledge of how to structure the order of autocorrelation dependencies (e.g., parents' genes will never be influenced by the genes of their children), it is impossible to tie the parameters and still guarantee an acyclic graph after roll out. Note that this is not a problem for undirected models such as RMNs or RDNs. Thus, RDNs enjoy the strengths of RBNs (namely, understandable knowledge representation and efficient learning) without being constrained by an acyclicity requirement.

An RMN for $\mathbf{X}$ uses a undirected model graph $U_M = (V_M, E_M)$ and a set of potential functions Φ to represent a joint distribution over $\mathbf{X}$. Again each node $v \in V_M$ corresponds to an $X_k^t \in \mathbf{X}$. RMNs use relational clique templates CT to specify the ways in which cliques are defined in U_M. Cliques templates are defined over a set of item types, a boolean constraint on how the types must relate to each other in the data graph, and a set of attributes to consider on the matching items. Let $C(CT)$

be the set of cliques in the graph U_M that match a specific clique template $C \in CT$. As with Markov networks, each clique $c \in C(CT)$ is associated with a set of variables X_c and a clique potential $\phi_c(x_c)$. Given (U_M, Φ), the joint probability for a set of values $\mathbf{x}$ is computed as a product over the clique templates in CT and the matches to the clique template c:

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{C_i \in CT} \prod_{c_j \in C(C_i)} \phi_{c_j}(x_{c_j}).$$

The RMN learning algorithm (Taskar et al., 2002) uses maximum-a-posteriori parameter estimation with Gaussian priors, modifying Markov network learning techniques. The algorithm assumes that the clique templates are pre-specified and thus does not search for the best structure. Because the user supplies a set of relational dependencies to consider (i.e., clique templates), it simply optimizes the potential functions for the specified templates.

RMNs are not hampered by an acyclicity constraint, so they can represent and reason with arbitrary forms of autocorrelation. This is particularly important for reasoning in relational data sets where autocorrelation dependencies are nearly ubiquitous and often cannot be structured in an acyclic manner. However, the tradeoff for this increased representational capability is a decrease in learning efficiency. Instead of closed-form parameter estimation, RMNs are trained with conjugate gradient methods, where each iteration requires a round of inference. In large cyclic relational inference graphs, the cost of inference is prohibitively expensive—in particular, without approximations to increase efficiency, feature selection is intractable.

Similar to the comparison with RBNs, RDNs enjoy the strengths of RMNs but not their weaknesses. More specifically, RDNs are able to reason with arbitrary forms of autocorrelation without being limited by efficiency concerns during learning. In fact, the pseudolikelihood estimation technique used by RDNs has been used recently to make feature selection tractable for conditional random fields (McCallum, 2003) and Markov logic networks (Kok and Domingos, 2005).

5.2 Probabilistic Logic Models

A second class of models for density estimation consists of extensions to conventional logic programming that support probabilistic reasoning in first-order logic environments. We will refer to this class of models as *probabilistic logic models* (PLMs). Examples of PLMs include Bayesian logic programs (Kersting and Raedt, 2002) and Markov logic networks (Richardson and Domingos, 2006).

PLMs represent a joint probability distribution over the ground atoms of a first-order knowledge base. The first-order knowledge base contains a set of first-order formulae, and the PLM associates a set of weights/probabilities with each of the formulae. Combined with a set of constants representing objects in the domain, PLMs specify a probability distribution over possible truth assignments to ground atoms of the first-order formulae. Learning a PLM consists of two tasks: generating the relevant first-order clauses, and estimating the weights/probabilities associated with each clause.

Within this class of models, Markov logic networks (MLNs) are most similar in nature to RDNs. In MLNs, each node is a grounding of a predicate in a first-order knowledge base, and features correspond to first-order formulae and their truth values. An MLN, unrolled over a set of objects in the domain, specifies an undirected Markov network. In this sense, they share the same strengths and weaknesses as RMNs—they are capable of representing cyclic autocorrelation relationships but suffer from decreased efficiency if full joint estimation is used during learning.

Recent work on structure learning for MLNs (Kok and Domingos, 2005) has investigated the use of pseudolikelihood to increase learning efficiency. The MLN structure learning algorithm restricts the search space by limiting the number of distinct variables in a clause, and then uses beam search, with a weighted pseudolikelihood scoring function, to find the best clauses to add to the network.

Although both RDNs and MLNs use pseudolikelihood techniques to learn model structures efficiently, they each employ a different representational formalism. In particular, MLNs use weighted logic formulae while RDNs use aggregate features in CPDs. Our future work will investigate the performance tradeoffs between RDN and MLN representations when combined with pseudolikelihood estimation.

5.3 Collective Inference

Collective inference models exploit autocorrelation dependencies in a network of objects to improve predictions. Joint relational models, such as those discussed above, are able to exploit autocorrelation to improve predictions by estimating joint probability distributions over the entire graph and collectively inferring the labels of related instances.

An alternative approach to collective inference combines local individual classification models (e.g., RBCs) with a joint inference procedure (e.g., relaxation labeling). Examples of this technique include iterative classification (Neville and Jensen, 2000), link-based classification (Lu and Getoor, 2003), and probabilistic relational neighbor models (Macskassy and Provost, 2003, 2004). These approaches to collective inference were developed in an ad hoc procedural fashion, motivated by the observation that they appear to work well in practice. RDNs formalize this approach in a principled framework—learning models locally (maximizing pseudolikelihood) and combining them with a global inference procedure (Gibbs sampling) to recover a full joint distribution. In this work we have demonstrated that autocorrelation is the reason behind improved performance in collective inference (see Jensen et al., 2004, for more detail) and explored the situations under which we can expect this type of approximation to perform well.

6. Discussion and Future Work

In this paper, we presented relational dependency networks, a new form of probabilistic relational model. We showed the RDN learning algorithm to be a relatively simple method for learning the structure and parameters of a probabilistic graphical model. In addition, RDNs allow us to exploit existing techniques for learning conditional probability distributions of relational data sets. Here we have chosen to exploit our prior work on RPTs, which construct parsimonious models of relational data, and RBCs, which are simple and surprisingly effective non-selective models. We expect the general properties of RDNs to be retained if other approaches to learning conditional probability distributions are used, given that those approaches learn accurate local models.

The primary advantage of RDNs is the ability to efficiently learn and reason with autocorrelation. Autocorrelation is a nearly ubiquitous phenomenon in relational data sets and the resulting dependencies are often cyclic in nature. If a data set exhibits autocorrelation, an RDN can learn the associated dependencies and then exploit those dependencies to improve overall inferences by collectively inferring values for the entire set of instances simultaneously. The real and synthetic data experiments in this paper show that collective inference with RDNs can offer significant improvement over conditional approaches when autocorrelation is present in the data. Except in rare cases, the performance of RDNs approaches the performance that would be possible if all the class labels of related instances were known. Furthermore, our experiments show that inference with RDNs is comparable, or superior, to RMN inference over a range of conditions, which indicates that pseudolikelihood estimation can be used effectively to learn an approximation of the full joint distribution.

We also presented learned RDNs for a number of real-world relational domains, demonstrating another strength of RDNs—their understandable and intuitive knowledge representation. Comprehensible models are a cornerstone of the knowledge discovery process, which seeks to identify novel and interesting patterns in large data sets. Domain experts are more willing to trust, and make regular use of, understandable models—particularly when the induced models are used to support additional reasoning. Understandable models also aid analysts' assessment of the utility of the additional relational information, potentially reducing the cost of information gathering and storage and the need for data transfer among organizations—increasing the practicality and feasibility of relational modeling.

Future work will compare RDNs to Markov logic networks in order to evaluate the performance tradeoffs for using pseudolikelihood approximations with different relational representations. Also, because our analysis indicates that each model reacts differently to the amount of seed information and level of autocorrelation, future work will attempt to quantify these effects on performance more formally. In particular, we are developing a bias/variance framework to decompose model errors into components of both the learning and inference processes (Neville and Jensen, 2006). Conventional bias/variance analysis is a useful tool for investigating the performance of machine learning algorithms, but it decomposes loss into errors due to aspects of the learning process alone. In relational and network applications, the collective inference process introduces an additional source of error—both through the use of approximate inference algorithms and through variation in the availability of test set information. Our framework can be used to evaluate hypotheses regarding the mechanisms behind poor model performance (e.g., identifiability problems increase RDN variance) and investigate algorithmic modifications designed to improve model performance.

Acknowledgments

The authors acknowledge the invaluable assistance of A. Shapira, as well as helpful comments from C. Loiselle and two anonymous reviewers.

This effort is supported by DARPA and NSF under contract numbers IIS0326249 and HR0011-04-1-0013. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied of DARPA, NSF, or the U.S. Government.

Appendix A. Synthetic Data Generation Models

We detail the manually specified conditional models used in Section 4.1. Specifically, there are three RBC and three RPT models. Each one is designed to generate data with low, medium, or high levels of autocorrelation ($0.25, 0.50, 0.75$).

$$
\begin{aligned}
RBC :=\quad & p(X_1|\mathbf{X_{1}}_R, X_2) \propto \prod_R p(X_{1R}|X_1)\cdot p(X_2|X_1).\\
RBC_{0.25}:\quad & p(X_{1R}|X_1) := p(X_{1R}=1|X_1=1) = p(X_{1R}=0|X_1=0) = 0.5625,\\
& p(X_2|X_1) := p(X_2=1|X_1=1) = p(X_2=0|X_1=0) = 0.75.\\
RBC_{0.50}:\quad & p(X_{1R}|X_1) := p(X_{1R}=1|X_1=1) = p(X_{1R}=0|X_1=0) = 0.6250,\\
& p(X_2|X_1) := p(X_2=1|X_1=1) = p(X_2=0|X_1=0) = 0.75.\\
RBC_{0.75}:\quad & p(X_{1R}|X_1) := p(X_{1R}=1|X_1=1) = p(X_{1R}=0|X_1=0) = 0.6875\\
& p(X_2|X_1) := p(X_2=1|X_1=1) = p(X_2=0|X_1=0) = 0.75.
\end{aligned}
$$

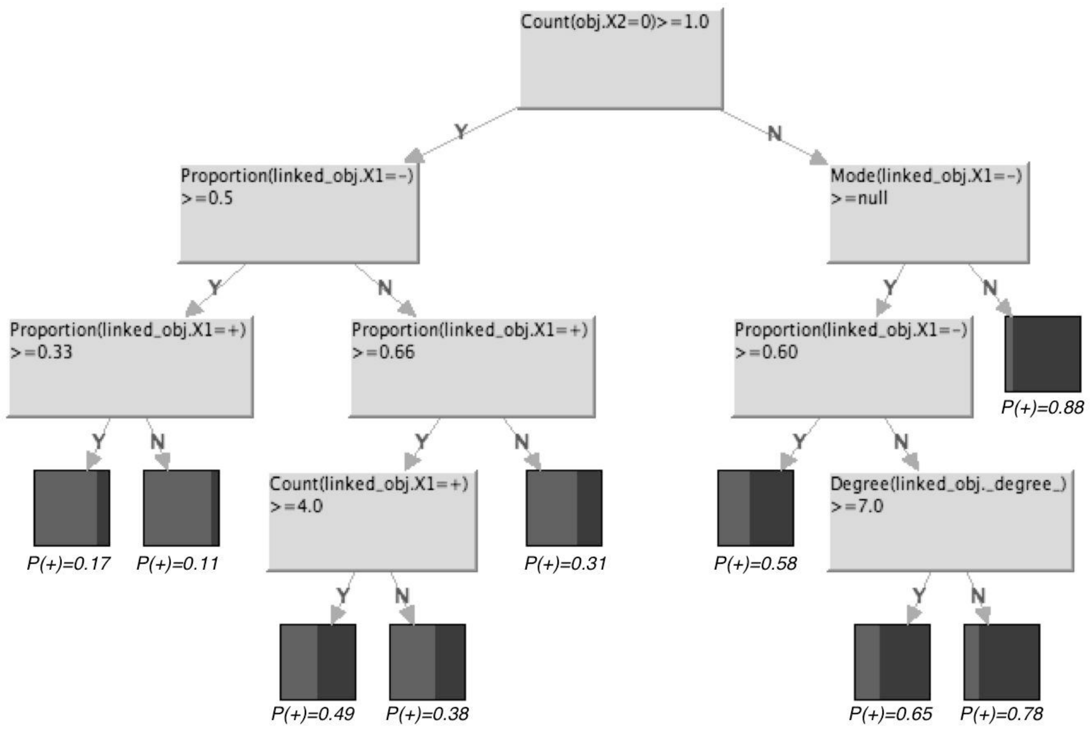

Count(obj.X2=0)>=1.0
Y
N
Proportion(linked_obj.X1=-) >=0.5
Mode(linked_obj.X1=-) >=null
Proportion(linked_obj.X1=+) >=0.33
Proportion(linked_obj.X1=+) >=0.66
Proportion(linked_obj.X1=-) >=0.60
P(+)=0.88
P(+)=0.17
P(+)=0.11
Count(linked_obj.X1=+) >=4.0
P(+)=0.31
P(+)=0.58
Degree(linked_obj._degree_) >=7.0
P(+)=0.49
P(+)=0.38
P(+)=0.65
P(+)=0.78

Figure 16: $RPT_{0.25}$ model used for synthetic data generation with low autocorrelation.

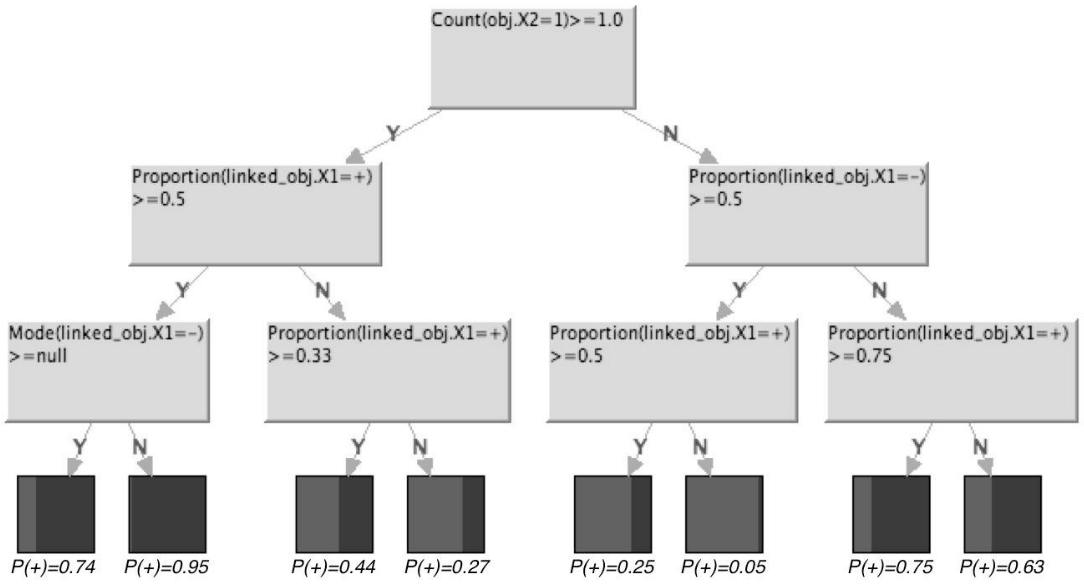

Count(obj.X2=1)>=1.0
Y
N
Proportion(linked_obj.X1=+) >=0.5
Proportion(linked_obj.X1=-) >=0.5
Mode(linked_obj.X1=-) >=null
Proportion(linked_obj.X1=+) >=0.33
Proportion(linked_obj.X1=+) >=0.5
Proportion(linked_obj.X1=+) >=0.75
P(+)=0.74
P(+)=0.95
P(+)=0.44
P(+)=0.27
P(+)=0.25
P(+)=0.05
P(+)=0.75
P(+)=0.63

Figure 17: $RPT_{0.50}$ model used for synthetic data generation with medium autocorrelation.

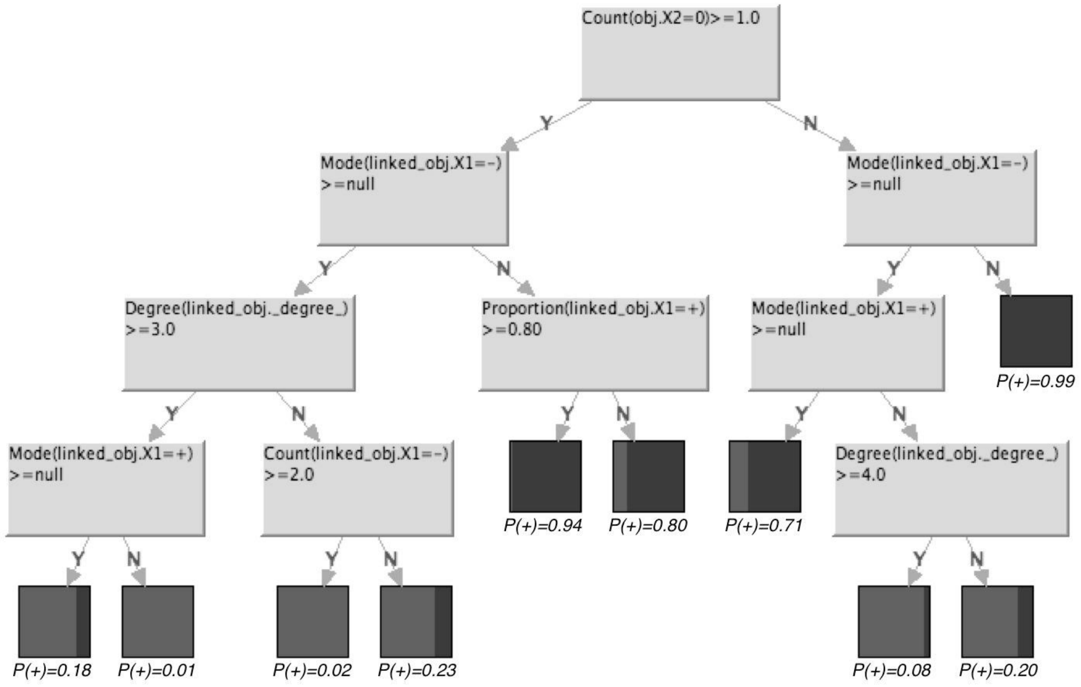

Count(obj.X2=0)>=1.0
Y
N
Mode(linked_obj.X1=-) >=null
Mode(linked_obj.X1=-) >=null
Degree(linked_obj._degree_) >=3.0
Proportion(linked_obj.X1=+) >=0.80
Mode(linked_obj.X1=+) >=null
P(+)=0.99
Mode(linked_obj.X1=+) >=null
Count(linked_obj.X1=-) >=2.0
P(+)=0.94
P(+)=0.80
P(+)=0.71
Degree(linked_obj._degree_) >=4.0
P(+)=0.18
P(+)=0.01
P(+)=0.02
P(+)=0.23
P(+)=0.08
P(+)=0.20

Figure 18: $RPT_{0.75}$ model used for synthetic data generation with high autocorrelation levels.

References

A. Bernstein, S. Clearwater, and F. Provost. The relational vector-space model and industry classification. In *Proceedings of the IJCAI-2003 Workshop on Learning Statistical Models from Relational Data*, pages 8–18, 2003.

J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society, Series B*, 36(2):192–236, 1974.

J. Besag. Statistical analysis of non-lattice data. *The Statistician*, 24(3):179–195, 1975.

H. Blau, N. Immerman, and D. Jensen. A visual query language for relational knowledge discovery. Technical Report 01-28, University of Massachusetts Amherst, Computer Science Department, 2001.

G. Casella and R. Berger. *Statistical Inference*. Duxbury, 2002.

S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 307–318, 1998.

F. Comets. On consistency of a class of estimators for exponential families of Markov random fields on the lattice. *The Annals of Statistics*, 20(1):455–468, 1992.

C. Cortes, D. Pregibon, and C. Volinsky. Communities of interest. In *Proceedings of the 4th International Symposium of Intelligent Data Analysis*, pages 105–114, 2001.

M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to extract symbolic knowledge from the World Wide Web. In *Proceedings of the 15th National Conference on Artificial Intelligence*, pages 509–516, 1998.

P. Domingos and M. Richardson. Mining the network value of customers. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 57–66, 2001.

T. Fawcett and F. Provost. Adaptive fraud detection. *Data Mining and Knowledge Discovery*, 1(3): 291–316, 1997.

P. Flach and N. Lachiche. 1BC: A first-order Bayesian classifier. In *Proceedings of the 9th International Conference on Inductive Logic Programming*, pages 92–103, 1999.

N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 1300–1309, 1999.

S. Geman and C. Graffine. Markov random field image models and their applications to computer vision. In *Proceedings of the 1986 International Congress of Mathematicians*, pages 1496–1517, 1987.

L. Getoor, N. Friedman, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Relational Data Mining*, pages 307–335. Springer-Verlag, 2001.

B. Gidas. Consistency of maximum likelihood and pseudolikelihood estimators for Gibbs distributions. In *Proceedings of the Workshop on Stochastic Differential Systems with Applications in Electrical/Computer Engineering, Control Theory, and Operations Research*, pages 129–145, 1986.

D. Heckerman, D. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering and data visualization. *Journal of Machine Learning Research*, 1:49–75, 2000.

D. Heckerman, C. Meek, and D. Koller. Probabilistic models for relational data. Technical Report MSR-TR-2004-30, Microsoft Research, 2004.

S. Hill, F. Provost, and C. Volinsky. Network-based marketing: Identifying likely adopters via consumer networks. *Statistical Science*, 22(2), 2006.

M. Jaeger. Relational Bayesian networks. In *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence*, pages 266–273, 1997.

D. Jensen and J. Neville. Linkage and autocorrelation cause feature selection bias in relational learning. In *Proceedings of the 19th International Conference on Machine Learning*, pages 259–266, 2002.

D. Jensen and J. Neville. Avoiding bias when aggregating relational data with degree disparity. In *Proceedings of the 20th International Conference on Machine Learning*, pages 274–281, 2003.

D. Jensen, J. Neville, and B. Gallagher. Why collective inference improves relational classification. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 593–598, 2004.

K. Kersting. Representational power of probabilistic-logical models: From upgrading to downgrading. In *IJCAI-2003 Workshop on Learning Statistical Models from Relational Data*, pages 61–62, 2003.

K. Kersting and L. De Raedt. Basic principles of learning Bayesian logic programs. Technical Report 174, Institute for Computer Science, University of Freiburg, 2002.

S. Kok and P. Domingos. Learning the structure of Markov logic networks. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 441–448, 2005.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289, 2001.

S. Lauritzen and N. Sheehan. Graphical models for genetic analyses. *Statistical Science*, 18(4): 489–514, 2003.

E. Lehmann and G. Casella. *Theory of Point Estimation*. Springer-Verlag, New York, 1998.

Q. Lu and L. Getoor. Link-based classification. In *Proceedings of the 20th International Conference on Machine Learning*, pages 496–503, 2003.

S. Macskassy and F. Provost. A simple relational classifier. In *Proceedings of the 2nd Workshop on Multi-Relational Data Mining, KDD2003*, pages 64–76, 2003.

S. Macskassy and F. Provost. Classification in networked data: A toolkit and a univariate case study. Technical Report CeDER-04-08, Stern School of Business, New York University, 2004.

A. McCallum. Efficiently inducing features of conditional random fields. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence*, pages 403–410, 2003.

A. McCallum, K. Nigam, J. Rennie, and K. Seymore. A machine learning approach to building domain-specific search engines. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 662–667, 1999.

K. Murphy, Y. Weiss, and M. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 467–479, 1999.

R. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Dept of Computer Science, University of Toronto, 1993.

J. Neville, O. Şimşek, D. Jensen, J. Komoroske, K. Palmer, and H. Goldberg. Using relational knowledge discovery to prevent securities fraud. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 449–458, 2005.

J. Neville and D. Jensen. Iterative classification in relational data. In *Proceedings of the Workshop on Statistical Relational Learning, 17th National Conference on Artificial Intelligence*, pages 42–49, 2000.

J. Neville and D. Jensen. Supporting relational knowledge discovery: Lessons in architecture and algorithm design. In *Proceedings of the Data Mining Lessons Learned Workshop, ICML2002*, pages 57–64, 2002.

J. Neville and D. Jensen. Collective classification with relational dependency networks. In *Proceedings of the 2nd Multi-Relational Data Mining Workshop, KDD2003*, pages 77–91, 2003.

J. Neville and D. Jensen. Dependency networks for relational data. In *Proceedings of the 4th IEEE International Conference on Data Mining*, pages 170–177, 2004.

J. Neville and D. Jensen. Bias/variance analysis for network data. In *Proceedings of the Workshop on Statistical Relational Learning, 23rd International Conference on Machine Learning*, 2006.

J. Neville, D. Jensen, L. Friedland, and M. Hay. Learning relational probability trees. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 625–630, 2003a.

J. Neville, D. Jensen, and B. Gallagher. Simple estimators for relational Bayesian classifers. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pages 609–612, 2003b.

C. Perlich and F. Provost. Aggregation-based feature invention and relational concept classes. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 167–176, 2003.

A. Popescul, L. Ungar, S. Lawrence, and D. Pennock. Statistical relational learning for document mining. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pages 275–282, 2003.

M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62:107–136, 2006.

S. Sanghai, P. Domingos, and D. Weld. Dynamic probabilistic relational models. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 992–1002, 2003.

B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, pages 485–492, 2002.

B. Taskar, E. Segal, and D. Koller. Probabilistic classification and clustering in relational data. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, pages 870–878, 2001.

H. White. *Estimation, Inference and Specification Analysis*. Cambridge University Press, New York, 1994.

B. Zadrozny and C. Elkan. Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. In *Proceedings of the 18th International Conference on Machine Learning*, pages 609–616, 2001.

Journal of Machine Learning Research 8 (2007) 693-723 Submitted 5/06; Published 3/07

Dynamic Conditional Random Fields: Factorized Probabilistic Models for Labeling and Segmenting Sequence Data

Charles Sutton CASUTTON@CS.UMASS.EDU
Andrew McCallum MCCALLUM@CS.UMASS.EDU
Khashayar Rohanimanesh[*] KHASH@CS.UMASS.EDU
Department of Computer Science
University of Massachusetts
140 Governors Drive
Amherst, Massachusetts 01003, USA

Editor: Michael Collins

Abstract

In sequence modeling, we often wish to represent complex interaction between labels, such as when performing multiple, cascaded labeling tasks on the same sequence, or when long-range dependencies exist. We present *dynamic conditional random fields (DCRFs)*, a generalization of linear-chain conditional random fields (CRFs) in which each time slice contains a set of state variables and edges—a distributed state representation as in dynamic Bayesian networks (DBNs)—and parameters are tied across slices. Since exact inference can be intractable in such models, we perform approximate inference using several schedules for belief propagation, including tree-based reparameterization (TRP). On a natural-language chunking task, we show that a DCRF performs better than a series of linear-chain CRFs, achieving comparable performance using only half the training data. In addition to maximum conditional likelihood, we present two alternative approaches for training DCRFs: *marginal likelihood training*, for when we are primarily interested in predicting only a subset of the variables, and *cascaded training*, for when we have a distinct data set for each state variable, as in transfer learning. We evaluate marginal training and cascaded training on both synthetic data and real-world text data, finding that marginal training can improve accuracy when uncertainty exists over the latent variables, and that for transfer learning, a DCRF trained in a cascaded fashion performs better than a linear-chain CRF that predicts the final task directly.

Keywords: conditional random fields, graphical models, sequence labeling

1. Introduction

The problem of labeling and segmenting sequences of observations arises in many different areas, including bioinformatics, music modeling, computational linguistics, speech recognition, and information extraction. Dynamic Bayesian Networks (DBNs) (Dean and Kanazawa, 1989; Murphy, 2002) are a popular method for probabilistic sequence modeling, because they exploit structure in the problem to compactly represent distributions over multiple state variables. Hidden Markov Models (HMMs), an important special case of DBNs, are a classical method for speech recognition (Rabiner, 1989) and part-of-speech tagging (Manning and Schütze, 1999). More complex DBNs have been used for applications as diverse as robot navigation (Theocharous et al., 2001), audio-visual speech recognition (Nefian et al., 2002), activity recognition (Bui et al., 2002), information

[*]. Current affiliation: Massachusetts Institute of Technology, Cambridge, MA 02139, USA

extraction (Skounakis et al., 2003; Peshkin and Pfeffer, 2003), and automatic speech recognition (Bilmes, 2003).

DBNs are typically trained to maximize the joint probability distribution $p(\mathbf{y},\mathbf{x})$ of a set of observation sequences $\mathbf{x}$ and labels $\mathbf{y}$. However, when the task does not require the ability to generate $\mathbf{x}$, such as in segmenting and labeling, modeling the joint distribution is a waste of modeling effort. Furthermore, generative models often must make problematic independence assumptions among the observed nodes in order to achieve tractability. In modeling natural language, for example, we may wish to use features of a word such as its identity, capitalization, prefixes and suffixes, neighboring words, membership in domain-specific lexicons, and category in semantic databases like WordNet—features which have complex interdependencies. Generative models that represent these interdependencies are in general intractable; but omitting such features or modeling them as independent has been shown to hurt accuracy (McCallum et al., 2000).

A solution to this problem is to model instead the conditional probability distribution $p(\mathbf{y}|\mathbf{x})$. The random vector $\mathbf{x}$ can include arbitrary, non-independent, domain-specific feature variables. Because the model is conditional, the dependencies among the features in $\mathbf{x}$ do not need to be explicitly represented. Conditionally-trained models have been shown to perform better than generatively-trained models on many tasks, including document classification (Taskar et al., 2002), part-of-speech tagging (Ratnaparkhi, 1996), extraction of data from tables (Pinto et al., 2003), segmentation of FAQ lists (McCallum et al., 2000), and noun-phrase segmentation (Sha and Pereira, 2003).

Conditional random fields (CRFs) (Lafferty et al., 2001; Sutton and McCallum, 2006) are undirected graphical models of the conditional distribution $p(\mathbf{y}|\mathbf{x})$. Early work on CRFs focused on the linear-chain structure (depicted in Figure 1) in which a first-order Markov assumption is made among labels. This model structure is analogous to conditionally-trained HMMs, and has efficient exact inference algorithms. Often, however, we wish to represent more complex interaction between labels—for example, when longer-range dependencies exist between labels, when the state can be naturally represented as a vector of variables, or when performing multiple cascaded labeling tasks on the same input sequence.

In this paper, we introduce *dynamic CRFs (DCRFs)*, which are a generalization of linear-chain CRFs that repeat structure and parameters over a sequence of state vectors. This allows us to both represent distributed hidden state and complex interaction among labels, as in DBNs, and to use rich, overlapping feature sets, as in conditional models. For example, the factorial structure in Figure 1(b), which we call a *factorial CRF (FCRF)*, includes links between cotemporal labels, explicitly modeling limited probabilistic dependencies between two different label sequences. Other types of DCRFs can model higher-order Markov dependence between labels (Figure 2), or incorporate a fixed-size memory. For example, a DCRF for part-of-speech tagging could include for each word a hidden state that is true if any previous word has been tagged as a verb.

Any DCRF with multiple state variables can be collapsed into a linear-chain CRF whose state space is the cross-product of the outcomes of the original state variables. However, such a linear-chain CRF needs exponentially many parameters in the number of variables. Like DBNs, DCRFs represent the joint distribution with fewer parameters by exploiting conditional independence relations.

In natural-language processing, DCRFs are especially attractive because they are a probabilistic generalization of cascaded, weighted finite-state transducers (Mohri et al., 2002). In general, many sequence-processing problems are traditionally solved by chaining errorful subtasks, such as chains of finite state transducers. In such an approach, however, errors early in processing nearly always cascade through the chain, causing errors in the final output. This problem can be solved by jointly representing the subtasks in a single graphical model, both explicitly representing their dependence, and preserving uncertainty between them. DCRFs can represent dependence between subtasks solved using finite-state transducers, such as phonological and morphological analysis, POS tagging, shallow parsing, and information extraction.

More specifically, in information extraction and data mining, McCallum and Jensen (2003) argue that the same kind of probabilistic unification can potentially be useful, because in many cases, we wish to mine a database that has been extracted from raw text. A unified probabilistic model for extraction and mining can allow data mining to take into account the uncertainty in the extraction, and allow extraction to benefit from emerging patterns produced by data mining. The applications here, in which DCRFs are used to jointly perform multiple sequence labeling tasks, can be viewed as an initial step toward that goal.

In this paper, we evaluate DCRFs on several natural-language processing tasks. First, a factorial CRF that learns to jointly predict parts of speech and segment noun phrases performs better than cascaded models that perform the two tasks in sequence. Also, we compare several schedules for belief propagation, showing that although exact inference is feasible, on this task approximate inference has lower total training time with no loss in testing accuracy.

In addition to conditional maximum likelihood training, we present two alternative training methods for DCRFs: *marginal training* and *cascaded training*. First, in some situations, we are primarily interested in predicting a few output variables $\mathbf{y}$, and the other output variables $\mathbf{w}$ are included only to help in modeling $\mathbf{y}$. For example, part-of-speech labels are usually never interesting in themselves, but rather are used to aid prediction of some higher level task. Training to maximize the joint conditional likelihood $p(\mathbf{y},\mathbf{w}|\mathbf{x})$ may then be inappropriate, because it may be forced to trade off accuracy among the $\mathbf{y}$, which we are primarily interested in, to obtain higher accuracy among $\mathbf{w}$, which we do not care about. For such situations, we explore the idea of training a DCRF by the marginal likelihood $\log p(\mathbf{y}|\mathbf{x}) = \log \sum_{\mathbf{w}} p(\mathbf{y},\mathbf{w}|\mathbf{x})$. On a natural-language chunking task, marginal training leads to a slight but not significant increase in accuracy on $\mathbf{y}$. We explain this unexpected result by further exploration on synthetic data, where we find that marginal training tends to improve performance most when the model has large uncertainty among the $\mathbf{y}$ labels, which is not the case in the chunking task.

Second, in other situations, a single fully-labeled data set is not available, and instead the outputs are partitioned into sets $(\mathbf{y}_0,\mathbf{y}_1,\dots,\mathbf{y}_\ell)$, and we have one data set D_0 labeled for $\mathbf{y}_0$, another data set D_1 labeled for $\mathbf{y}_1$, and so on. For example, this can be the case in *transfer learning*, in which we wish to use previous learning problems (that is, $\mathbf{y}_0,\mathbf{y}_1,\dots,\mathbf{y}_{\ell-1}$) to improve performance on a new task $\mathbf{y}_\ell$. To train a DCRF without a single fully-labeled training set, we propose a *cascaded training* procedure, in which first we train a CRF p_0 to predict $\mathbf{y}_0$ on D_0, then we annotate D_1 with the most likely prediction from p_0, then we train a CRF p_1 on $p(\mathbf{y}_1|\mathbf{y}_0,\mathbf{x})$, and finally, at test time, perform inference jointly in the resulting DCRF. Compared to other work in transfer learning, an interesting aspect of this approach is that the model includes no shared latent structure between subtasks; rather, the probabilistic dependence between tasks is modeled directly. On a benchmark information extraction task, we show that a DCRF trained in a cascaded fashion performs better than a linear-chain CRF on the target task.

The rest of the paper is structured as follows. In Section 2, we describe the general framework of CRFs. Then, in Section 3, we define DCRFs, and explain methods for approximate inference and parameter estimation, including exact conditional maximum likelihood (Section 3.3.1), approximate

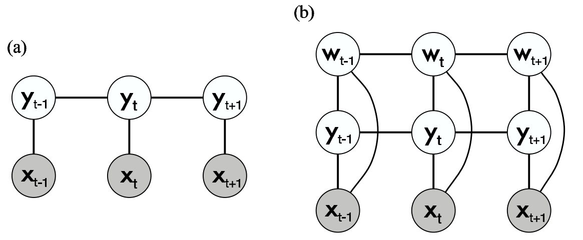


Figure 1: Graphical representation of (a) linear-chain CRF, and (b) factorial CRF. Although the hidden nodes can depend on observations at any time step, for clarity we have shown links only to observations at the same time step.

parameter estimation using BP (Section 3.3.2), and cascaded parameter estimation (Section 3.3.3). Then, in Section 3.4, we describe inference and parameter estimation in marginal DCRFs. In Section 4, we present the experimental results, including evaluation of factorial CRFs on noun-phrase chunking (Section 4.1), comparison of BP schedules in FCRFs (Section 4.2), evaluation of marginal DCRFs on both the chunking data and synthetic data (Section 4.3), and cascaded training of DCRFs for transfer learning (Section 4.4). Finally, in Section 5 and Section 6, we present related work and conclude.

2. Conditional Random Fields (CRFs)

Conditional random fields (CRFs) (Lafferty et al., 2001; Sutton and McCallum, 2006) are conditional probability distributions that factorize according to an undirected model. CRFs are defined as follows. Let $\mathbf{y}$ be a set of output variables that we wish to predict, and $\mathbf{x}$ be a set of input variables that are observed. For example, in natural language processing, $\mathbf{x}$ may be a sequence of words $\mathbf{x} = \{x_t\}$ for $t = 1, \dots, T$ and $\mathbf{y} = \{y_t\}$ a sequence of labels. Let $\mathcal{G}$ be a factor graph over $\mathbf{y}$ and $\mathbf{x}$ with factors $C = \{\Phi_c(\mathbf{y}_c, \mathbf{x}_c)\}$, where $\mathbf{x}_c$ is the set of input variables that are arguments to the local function Φ_c, and similarly for $\mathbf{y}_c$. A *conditional random field* is a conditional distribution p_Λ that factorizes as

$$p_\Lambda(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{c \in C} \Phi_c(\mathbf{y}_c, \mathbf{x}_c),$$

where $Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_{c \in C} \Phi_c(\mathbf{y}_c, \mathbf{x}_c)$ is a normalization factor over all state sequences for the sequence $\mathbf{x}$. We assume the potentials factorize according to a set of features $\{f_k\}$, as

$$\Phi_c(\mathbf{y}_c, \mathbf{x}_c) = \exp\left(\sum_k \lambda_k f_k(\mathbf{y}_c, \mathbf{x}_c)\right),$$

so that the family of distributions $\{p_\Lambda\}$ is an exponential family. In this paper, we shall assume that the features are given and fixed. The model parameters are a set of real weights $\Lambda = \{\lambda_k\}$, one weight for each feature.

Many previous applications use the *linear-chain CRF*, in which a first-order Markov assumption is made on the hidden variables. A graphical model for this is shown in Figure 1. In this case, the cliques of the conditional model are the nodes and edges, so that there are feature functions

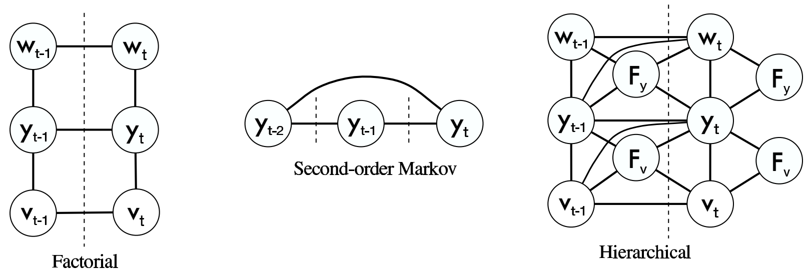


Figure 2: Examples of DCRFs. The dashed lines indicate the boundary between time steps. The input variables $\mathbf{x}$ are not shown.

$f_k(y_{t-1}, y_t, \mathbf{x}, t)$ for each label transition. (Here we write the feature functions as potentially depending on the entire input sequence.) Feature functions can be arbitrary. For example, a feature function $f_k(y_{t-1}, y_t, \mathbf{x}, t)$ could be a binary test that has value 1 if and only if y_{t-1} has the label "*adjective*", y_t has the label "*proper noun*", and x_t begins with a capital letter.

3. Dynamic CRFs

In this section, we define dynamic CRFs (Section 3.1) and describe methods for inference (Section 3.2) and parameter estimation (Section 3.3).

3.1 Model Representation

A dynamic CRF (DCRF) is a conditional distribution that factorizes according to an undirected graphical model whose structure and parameters are repeated over a sequence. As with a DBN, a DCRF can be specified by a template that gives the graphical structure, features, and weights for two time slices, which can then be unrolled given an input $\mathbf{x}$. The same set of features and weights is used at each sequence position, so that the parameters are tied across the network. Several example templates are given in Figure 2.

Now we give a formal description of the unrolling process. Let $\mathbf{y} = \{\mathbf{y}_1 \dots \mathbf{y}_T\}$ be a sequence of random vectors $\mathbf{y}_i = (y_{i1} \dots y_{im})$, where $\mathbf{y}_i$ is the state vector at time i, and y_{ij} is the value of variable j at time i. To give the likelihood equation for arbitrary DCRFs, we require a way to describe a clique in the unrolled graph independent of its position in the sequence. For this purpose we introduce the concept of a *clique index*. Given a time t, we can denote any variable $\mathbf{y}_{ij}$ in $\mathbf{y}$ by two integers: its index j in the state vector $\mathbf{y}_i$, and its time offset $\Delta t = i - t$. We will call a set $c = \{(\Delta t, j)\}$ of such pairs a clique index, which denotes a set of variables $\mathbf{y}_{t,c}$ by $\mathbf{y}_{t,c} \equiv \{y_{t+\Delta t, j} \,|\, (\Delta t, j) \in c\}$. That is, $\mathbf{y}_{t,c}$ is the set of variables in the unrolled version of clique index c at time t.

Now we can formally define DCRFs:

Definition 1 *Let C be a set of clique indices, $F = \{f_k(\mathbf{y}_{t,c}, \mathbf{x}, t)\}$ be a set of feature functions and $\Lambda = \{\lambda_k\}$ be a set of real-valued weights. Then the distribution p is a* dynamic conditional random field *if and only if*

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_t \prod_{c \in C} \exp\left(\sum_k \lambda_k f_k(\mathbf{y}_{t,c}, \mathbf{x}, t) \right)$$

where $Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_t \prod_{c \in C} \exp\left(\sum_k \lambda_k f_k(\mathbf{y}_{t,c}, \mathbf{x}, t)\right)$ *is the partition function.*

Although we define a DCRF has having the same set of features for all of its cliques, in practice we choose each feature function f_k so that it is non-zero only on cliques with some particular index c_k. Thus, we will sometimes think of each clique index has having its own set of features and weights, and speak of f_k and λ_k as having an associated clique index c_k.

DCRFs generalize not only linear-chain CRFs, but more complicated structures as well. For example, in this paper, we use a *factorial CRF (FCRF)*, which has linear chains of labels, with connections between cotemporal labels. We name these after factorial HMMs (Ghahramani and Jordan, 1997). Figure 1(b) shows an unrolled factorial CRF. Consider an FCRF with L chains, where $Y_{\ell,t}$ is the variable in chain ℓ at time t. The clique indices for this DCRF are of the form $\{(0,\ell),(1,\ell)\}$ for each of the within-chain edges and $\{(0,\ell),(0,\ell+1)\}$ for each of the between-chain edges. The FCRF p defines a distribution over output variables as:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \left(\prod_{t=1}^{T-1} \prod_{\ell=1}^{L} \Phi_\ell(y_{\ell,t}, y_{\ell,t+1}, \mathbf{x}, t) \right) \left(\prod_{t=1}^{T} \prod_{\ell=1}^{L-1} \Psi_\ell(y_{\ell,t}, y_{\ell+1,t}, \mathbf{x}, t) \right),$$

where $\{\Phi_\ell\}$ are the factors over the within-chain edges, $\{\Psi_\ell\}$ are the factors over the between-chain edges, and $Z(\mathbf{x})$ is the partition function. The factors are modeled using the features $\{f_k\}$ and weights $\{\lambda_k\}$ of G as:

$$\Phi_\ell(y_{\ell,t}, y_{\ell,t+1}, \mathbf{x}, t) = \exp\left\{ \sum_k \lambda_k f_k(y_{\ell,t}, y_{\ell,t+1}, \mathbf{x}, t) \right\},$$

$$\Psi_\ell(y_{\ell,t}, y_{\ell+1,t}, \mathbf{x}, t) = \exp\left\{ \sum_k \lambda_k f_k(y_{\ell,t}, y_{\ell+1,t}, \mathbf{x}, t) \right\}.$$

More complicated structures are also possible, such as second-order CRFs, and hierarchical CRFs, which are moralized versions of the hierarchical HMMs of Fine et al. (1998).[1] As in DBNs, this factorized structure can use many fewer parameters than the cross-product state space: even the two-level FCRF we discuss below uses less than an eighth of the parameters of the corresponding cross-product CRF.

3.2 Inference in DCRFs

Inference in a DCRF can be done using any inference algorithm for undirected models. For an unlabeled sequence $\mathbf{x}$, we typically wish to solve two inference problems: (a) computing the marginals $p(\mathbf{y}_{t,c}|\mathbf{x})$ over all cliques $\mathbf{y}_{t,c}$, and (b) computing the Viterbi decoding $\mathbf{y}^* = \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x})$. The Viterbi decoding can be used to label a new sequence, and marginal computation is used for parameter estimation (Section 3.3).

If the number of states is not large, the simplest approach is to form a linear chain whose output space is the cross-product of the original DCRF outputs, and then perform forward-backward. In

1. Hierarchical HMMs were shown to be DBNs by Murphy and Paskin (2001).

other words, a DCRF can always be viewed as a linear-chain CRF whose feature functions take a special form, analogous to the relationship between generative DBNs and HMMs. The cross-product space is often very large, however, in which case this approach is infeasible. Alternatively, one can perform exact inference by applying the junction tree algorithm to the unrolled DCRF, or by using the special-purpose inference algorithms that have been designed for DBNs (Murphy, 2002), which can avoid storing the full unrolled graph.

In complex DCRFs, though, exact inference can still be expensive, making approximate methods necessary. Furthermore, because marginal computation is needed during training, inference must be efficient so that we can use large training sets even if there are many labels. The largest experiment reported here required computing pairwise marginals in 866,792 different graphical models: one for each training example in each iteration of a convex optimization algorithm. In the remainder of the section, we describe approximate inference using loopy belief propagation (BP).

Although belief propagation is exact only in certain special cases, in practice it has been a successful approximate method for general graphical models (Murphy et al., 1999; Aji et al., 1998). For simplicity, we describe BP for a pairwise CRF with factors $\{\Phi(x_u, x_v)\}$, but the algorithm can be generalized to arbitrary factor graphs. Belief propagation algorithms iteratively update a vector $\mathbf{m} = (m_u(x_v))$ of messages between pairs of vertices x_u and x_v. The update from x_u to x_v is given by:

$$m_u(x_v) \leftarrow \sum_{x_u} \Phi(x_u, x_v) \prod_{x_t \neq x_v} m_t(x_u), \tag{1}$$

where $\Phi(x_u, x_v)$ is the potential on the edge (x_u, x_v). Performing this update for one edge (x_u, x_v) in one direction is called *sending a message* from x_u to x_v. Given a message vector $\mathbf{m}$, approximate marginals are computed as

$$p(x_u, x_v) \leftarrow \kappa \Phi(x_u, x_v) \prod_{x_t \neq x_v} m_t(x_u) \prod_{x_w \neq x_u} m_w(x_v),$$

where κ is a normalization constant.

At each iteration of belief propagation, messages can be sent in any order, and choosing a good schedule can affect how quickly the algorithm converges. We describe two schedules for belief propagation: tree-based and random. The tree-based schedule, also known as tree reparameterization (TRP) (Wainwright et al., 2001; Wainwright, 2002), propagates messages along a set of cross-cutting spanning trees of the original graph. At each iteration of TRP, a spanning tree $\mathcal{T}^{(i)} \in \Upsilon$ is selected, and messages are sent in both directions along every edge in $\mathcal{T}^{(i)}$, which amounts to exact inference on $\mathcal{T}^{(i)}$. In general, trees may be selected from any set $\Upsilon = \{\mathcal{T}\}$ as long as the trees in Υ cover the edge set of the original graph. In practice, we select trees randomly, but we select first edges that have never been used in any previous iteration.

The random schedule simply sends messages across all edges in random order. To improve convergence, we arbitrarily order each edge $e_i = (s_i, t_i)$ and send all messages $m_{s_i}(t_i)$ before any messages $m_{t_i}(s_i)$. Note that for a graph with V nodes and E edges, TRP sends $O(V)$ messages per BP iteration, while the random schedule sends $O(E)$ messages.

An alternative to schedule is a synchronous schedule, in which conceptually all messages are sent at the same time. In the tree-based and random schedules, once a message is updated, its new values are immediately available for other messages. In the synchronous schedule, on the other hand, when computing a message $m_u^{(j)}(x_v)$ at iteration j of BP, the previous message values $m_t^{(j-1)}(x_u)$ are always used, even if an updated value $m_t^{(j)}(x_u)$ has been computed. We do not report results from the synchronous schedule in this paper, because preliminary experiments indicated that it requires many more iterations to converge than the other schedules.

Finally, dynamic schedules for BP (Elidan et al., 2006), which depend on the current message values during inference, have recently been shown to converge significantly faster than TRP on sufficiently difficult graphs, and may be preferable to TRP on certain DCRFs. We do not consider them in this paper, however.

To perform Viterbi decoding, we use the same propagation algorithms, except that the summation in Equation (1) is replaced by maximization.

3.3 Parameter Estimation in DCRFs

In this section, we discuss exact parameter estimation (Section 3.3.1) and approximate parameter estimation using belief propagation (Section 3.3.2). Also, we introduce a novel approximate training procedure called cascaded training (Section 3.3.3).

3.3.1 EXACT PARAMETER ESTIMATION

The parameter estimation problem is to find a set of parameters $\Lambda = \{\lambda_k\}$ given training data $\mathcal{D} = \{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^{N}$. More specifically, we optimize the conditional log-likelihood

$$\mathcal{L}(\Lambda) = \sum_i \log p_\Lambda(\mathbf{y}^{(i)} \mid \mathbf{x}^{(i)}).$$

The derivative of this with respect to a parameter λ_k associated with clique index c is

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \lambda_k} &= \sum_i \sum_t f_k(\mathbf{y}_{t,c}^{(i)}, \mathbf{x}^{(i)}, t) \\ &\quad - \sum_i \sum_t \sum_{\mathbf{y}_{t,c}} p_\Lambda(\mathbf{y}_{t,c} \mid \mathbf{x}^{(i)}) f_k(\mathbf{y}_{t,c}, \mathbf{x}^{(i)}, t).\end{aligned} \tag{2}$$

where $\mathbf{y}_{t,c}^{(i)}$ is the assignment to $\mathbf{y}_{t,c}$ in $\mathbf{y}^{(i)}$, and $\mathbf{y}_{t,c}$ ranges over assignments to the clique c. Observe that it is the factor $p_\Lambda(\mathbf{y}_{t,c} \mid \mathbf{x}^{(i)})$ that requires us to compute marginal probabilities in the unrolled DCRF.

To reduce overfitting, we define a prior $p(\Lambda)$ over parameters, and optimize $\log p(\Lambda|\mathcal{D}) = \mathcal{L}(\Lambda) + \log p(\Lambda)$. We use a spherical Gaussian prior with mean $\mu = 0$ and covariance matrix $\Sigma = \sigma^2 I$, so that the gradient becomes

$$\frac{\partial p(\Lambda|\mathcal{D})}{\partial \lambda_k} = \frac{\partial \mathcal{L}}{\partial \lambda_k} - \frac{\lambda_k}{\sigma^2}.$$

This corresponds to optimizing $\mathcal{L}(\Lambda)$ using ℓ_2 regularization. Other priors are possible, for example, an exponential prior, which corresponds to regularizing $\mathcal{L}(\Lambda)$ by an ℓ_1 norm.

The function $\log p(\Lambda|\mathcal{D})$ is convex, and can be optimized by any number of techniques, as in other maximum-entropy models (Lafferty et al., 2001; Berger et al., 1996). For example, Newton's method achieves fast convergence, but requires computing the Hessian, which is expensive both because computing the individual second derivatives is expensive, and because the Hessian is of size K^2, where K is the number of parameters. Typically in text applications, the number of parameters can be anywhere from the tens of thousands to the millions, so that maintaining a full $K \times K$ matrix is infeasible. Instead, we use quasi-Newton methods, which iteratively maintain an approximation to the Hessian using only the gradient. But standard quasi-Newton methods, such as BFGS, also approximate the Hessian by a full $K \times K$ matrix, which is too large. Therefore, we use a limited-memory version of BFGS, called L-BFGS (Nocedal and Wright, 1999), which approximates the Hessian in such a way that the full $K \times K$ matrix is never calculated explicitly. L-BFGS has previously been shown to outperform other optimization algorithms for linear-chain CRFs (Sha and Pereira, 2003; Malouf, 2002; Wallach, 2002). In particular, iterative scaling algorithms such as GIS and IIS have been shown to be much slower than gradient-based algorithms such as L-BFGS or preconditioned conjugate gradient.

All of the methods we have described are batch methods, meaning that they examine all of the training data before making a gradient update. Recently, stochastic gradient methods, which make gradient updates based on small subsets of the data, have been shown to converge significantly faster for linear-chain CRFs (Vishwanathan et al., 2006). It is likely that stochastic gradient methods would perform similarly well for DCRFs, but we do not use them in the experiments reported here.

The discussion above was for the fully-observed case, where the training data include observed values for all variables in the model. If some nodes are unobserved, the optimization problem becomes more difficult, because the log likelihood is no longer convex in general. We describe this case in Section 3.4.1.

3.3.2 APPROXIMATE PARAMETER ESTIMATION USING BP

For models in which exact inference is infeasible, we use approximate inference during training. In Section 3.2, we discussed inference in DCRFs. In this section, we discuss additional issues that arise when using BP during training. First, to simplify notation in this section, we will write a DCRF as $p(\mathbf{y}|\mathbf{x}) = Z(\mathbf{x})^{-1} \prod_t \prod_c \psi_{t,c}(\mathbf{y}_{t,c})$, where each factor in the unrolled DCRF is defined as

$$\psi_{t,c}(\mathbf{y}_{t,c}) = \exp\left\{\sum_k \lambda_k f_k(\mathbf{y}_{t,c}, \mathbf{x}, t)\right\}.$$

That is, we drop the dependence of the factors $\{\psi_{t,c}\}$ on $\mathbf{x}$.

For approximate parameter estimation, the basic procedure is to optimize using the gradient (2) as described in the last section, but instead of running an exact inference algorithm on each training example to obtain marginal distributions $p_\Lambda(\mathbf{y}_{t,c} \mid \mathbf{x}^{(i)})$, we run BP on each training instance to obtain approximate factor beliefs $b_{t,c}(\mathbf{y}_{t,c})$ for each clique $\mathbf{y}_{t,c}$ and approximate node beliefs $b_s(y_s)$ for each output variable s.

Now, although BP provides approximate marginal distributions that allow calculating the gradient, there is still the issue of how to calculate an approximate likelihood. In particular, we need an approximate objective function whose gradient is equal to the approximate gradient we have just described. We use the approximate likelihood

$$\hat{\ell}(\Lambda; \{b\}) = \sum_i \log\left[\frac{\prod_t \prod_c b_{t,c}(\mathbf{y}_{t,c}^{(i)})}{\prod_s b_s(y_s^{(i)})^{d_s - 1}}\right], \tag{3}$$

where s ranges over output variables (that is, components of $\mathbf{y}$), and d_s is the degree of s (that is, the number of factors $\psi_{c,t}$ that depend on the variable s). In other words, we approximate the joint likelihood by the product over each clique's approximate belief, dividing by the node beliefs to avoid overcounting. In the remainder of this section, we justify this choice.

BP can be viewed as attempting to solve an optimization problem over possible choices of marginal distributions, for a particular cost function called the *Bethe free energy*. More technically, it has been shown that fixed points of BP are stationary points of the Bethe free energy (for more details, see Yedidia et al., 2005), when minimized over locally-consistent beliefs. The Bethe energy is an approximation to another cost function, which Yedidia et al. call the Helmholtz free energy. Since the minimum Helmhotlz energy is exactly $-\log Z(\mathbf{x})$, we approximate $-\log Z(\mathbf{x})$ by the minimizing value of the Bethe energy, that is:

$$\ell_{\text{BETHE}}(\Lambda) = \sum_i \sum_t \sum_c \log \psi_{t,c}(\mathbf{y}_{t,c}) + \sum_i \min_{\{b\}} \mathcal{F}_{\text{BETHE}}(b), \tag{4}$$

where $\mathcal{F}_{\text{BETHE}}$ is the Bethe free energy, which is defined as

$$\mathcal{F}_{\text{BETHE}}(b) = \sum_t \sum_c \sum_{\mathbf{y}_{t,c}} b_{t,c}(\mathbf{y}_{t,c}) \log \frac{b_{t,c}(\mathbf{y}_{t,c})}{\psi_{t,c}(\mathbf{y}_{t,c})} - \sum_s (d_s - 1) \sum_{x_s} b_s(y_s) \log b_s(y_s).$$

So approximate training with BP can be viewed as solving a saddlepoint problem of maximizing ℓ_{BETHE} with respect to the model parameters and minimizing with respect to the beliefs $b_{t,c}(\mathbf{y}_{t,c})$. Approximate training using BP is just coordinate ascent: BP optimizes ℓ_{BETHE} with respect to b for fixed Λ; and a step along the gradient (2) optimizes ℓ_{BETHE} with respect to Λ for fixed b. Taking the partial derivative of (4) with respect to a weight λ_k, we obtain the gradient (2) with marginal distributions replaced by beliefs, as desired.

To justify the approximate likelihood (3), we note that the Bethe free energy can be written a dual form, in which the variables are interpreted as log messages rather than beliefs. Details of this are presented by Minka (2001a, 2005). Substituting the Bethe dual problem into (4) and simplifying yields (3).

3.3.3 CASCADED PARAMETER ESTIMATION

Joint maximum likelihood training assumes that we have access to data in which we have observed all of the variables. Sometimes this is not the case. One example is *transfer learning*, which is the general problem of using previous learning problems that a system has seen to aid its learning of new, related tasks. Usually in transfer learning, we have one data set labeled with the old task variables and one with the new task variables, but no data that is jointly labeled. In this section, we describe a cascaded parameter estimation procedure that can be applied when the data is not fully labeled.

For a factorial CRF with N levels, the basic idea is to train each level separately as if it were a linear-chain CRF, using the single-best prediction of the previous level as a feature. At the end, each set of individually-trained weights defines a pair of factors, which are simply multiplied together to form the full FCRF. The cascaded procedure is described formally in Algorithm 1. In this description, the two clique indices for each level ℓ of the FCRF are denoted by c_ℓ^{W} for the within-level cliques, with features $f_{\ell,k}^{\text{W}}(y_t^\ell, y_{t+1}^\ell, \mathbf{x}, t)$ and weights Λ_ℓ^{W}; and another index c_ℓ^{P} for the between-level cliques, with features $f_{\ell,k}^{\text{P}}(y_t^\ell, y_t^{\ell-1}, \mathbf{x}, t)$ and weights Λ_ℓ^{P}.

For simplicity, we have presented cascaded training for factorial CRFs, but it can be generalized to other DCRF structures, as long as the DCRF templates can be partitioned in a way that respects the available labels. In Section 4.4, we evaluate cascaded training on a transfer learning problem.

Algorithm 1 Cascaded training for Factorial CRFs

```
1: Train a linear-chain CRF on $\log p(\mathbf{y}_0|\mathbf{x})$, yielding weights $\Lambda_0^{\mathrm{W}}$.
2: for all levels $\ell$ do
3:    Compute Viterbi labeling $\mathbf{y}_{\ell-1}^* = \arg\max_{\mathbf{y}_{\ell-1}} p(\mathbf{y}_{\ell-1}|\mathbf{y}_{\ell-2}^*, \mathbf{x})$ for each training instance $i$.
4:    Train a linear-chain CRF to maximize $\log p(\mathbf{y}_\ell|\mathbf{y}_{\ell-1}^*, \mathbf{x})$, yielding weights $\Lambda_\ell^{\mathrm{W}}$ and $\Lambda_\ell^{\mathrm{P}}$.
5: end for
6: return factorial CRF defined as
```

$$p(\mathbf{y}|\mathbf{x}) \propto \prod_{\ell=0}^{N}\prod_{t=1}^{T}\Psi^{\mathrm{W}}(y_t^\ell, y_{t+1}^\ell, \mathbf{x}, t)\Psi^{\mathrm{P}}(y_t^\ell, y_t^{\ell-1}, \mathbf{x}, t)$$

where

$$\Psi^{\mathrm{W}}(y_t^\ell, y_{t+1}^\ell, \mathbf{x}, t) = \exp\{\sum_k \lambda_{k,\ell}^{\mathrm{W}} f_{\ell,k}^{\mathrm{W}}(y_t^\ell, y_{t+1}^\ell, \mathbf{x}, t)\},$$
$$\Psi^{\mathrm{P}}(y_t^\ell, y_t^{\ell-1}, \mathbf{x}, t) = \exp\{\sum_k \lambda_{k,\ell}^{\mathrm{P}} f_{\ell,k}^{\mathrm{P}}(y_t^\ell, y_t^{\ell-1}, \mathbf{x}, t)\}.$$

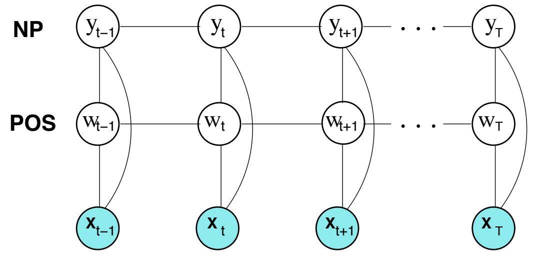


Figure 3: Graphical representation of factorial CRF for the joint noun phrase chunking and part-of-speech tagging problem, where the chain **y** represents the NP labels, and the chain **w** represents the POS labels.

3.4 Marginal DCRFs

In some applications, we are primarily interested in a few *main variables*, and other *auxiliary variables* are included in the model simply to aid in predicting the main variables. For example, in the chunking model of Section 4.1, the task is to predict the boundaries of noun phrases, and we are interested in predicting part-of-speech tags only insofar as they help to do this. In such applications, training by maximizing the joint likelihood might be inappropriate, because it might be forced to trade off modeling the main variables against modeling the other variables. This motivates the following idea: rather than modeling all of the variables jointly given the input, we can model the main variables only, marginalizing out the auxiliary variables. The idea is to let the model focus its effort on modeling the main variables, while retaining the useful information from the other variables. In this section, we discuss this model class, which we call the *marginal DCRF*. First, in Section 3.4.1, we define the model class and describe maximum likelihood parameter estimation. Then, in Section 3.4.2, we discuss inference in marginal DCRFs.

3.4.1 MARGINAL DCRFS, AND PARAMETER ESTIMATION

First, we define the marginal DCRF model.

Definition 2 *A distribution p is a* marginal DCRF *over random vectors* $(\mathbf{y}, \mathbf{w})$ *given inputs* $\mathbf{x}$ *if it can be written as:*

$$p_\Lambda(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{w}} p_\Lambda(\mathbf{y}, \mathbf{w}|\mathbf{x}),$$

where $p_\Lambda(\mathbf{y}, \mathbf{w}|\mathbf{x})$ *is a DCRF.*

Parameter estimation proceeds by optimizing the log likelihood

$$\mathcal{L}(\Lambda) = \sum_i \log p_\Lambda(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}) = \sum_i \log \sum_{\mathbf{w}} p_\Lambda(\mathbf{y}^{(i)}, \mathbf{w}|\mathbf{x}^{(i)}). \tag{5}$$

Intuitively, this objective function concentrates on the conditional likelihood over the variables $\mathbf{y}$, possibly sacrificing accuracy on the variables $\mathbf{w}$. Indeed, this objective function ignores any observations of $\mathbf{w}$ in the training set. We can take these observations into account in some partial way by careful choice of initialization, as we describe in Section 4.3.

The derivative of (5) with respect to a parameter λ_k associated with a clique index c is:

$$\begin{aligned}\frac{\partial \mathcal{L}(\Lambda)}{\partial \lambda_k} = &\sum_i \sum_t \sum_{\tilde{\mathbf{w}}_{t,c}} p(\tilde{\mathbf{w}}_{t,c}|\mathbf{y}^{(i)}, \mathbf{x}^{(i)}) f_k(\mathbf{y}^{(i)}_{t,c}, \tilde{\mathbf{w}}_{t,c}, \mathbf{x}^{(i)}_{t,c}) - \\ &\sum_i \sum_t \sum_{\tilde{\mathbf{y}}_{t,c}, \tilde{\mathbf{w}}_{t,c}} p(\tilde{\mathbf{y}}_{t,c}, \tilde{\mathbf{w}}_{t,c}|\mathbf{x}^{(i)}) f_k(\tilde{\mathbf{y}}_{t,c}, \tilde{\mathbf{w}}_{t,c}, \mathbf{x}^{(i)}),\end{aligned} \tag{6}$$

where $\mathbf{y}^{(i)}_{t,c}$ is the subset of the training vector $\mathbf{y}^{(i)}$ that occurs in clique c instantiated at time t; the summation over $\tilde{\mathbf{y}}_{t,c}$ ranges over all assignments to clique c; and $\mathbf{w}_{t,c}$ and $\tilde{\mathbf{w}}_{t,c}$ are defined similarly. Also, to reduce overfitting, we include a prior on parameters, as in Section 3.3.

Another way to understand the gradient is as follows. For any function $f(\lambda)$, we have

$$\frac{\partial f}{\partial \lambda} = f(\lambda) \frac{\partial \log f}{\partial \lambda},$$

which can be seen by applying the chain rule to $\log f$ and rearranging. Applying this to the marginal likelihood $\ell(\Lambda) = \log \sum_{\mathbf{w}} p(\mathbf{y}, \mathbf{w}|\mathbf{x})$, we get

$$\begin{aligned}\frac{\partial \ell}{\partial \lambda_k} &= \frac{1}{\sum_{\mathbf{w}} p(\mathbf{y}, \mathbf{w}|\mathbf{x})} \sum_{\mathbf{w}} \frac{\partial}{\partial \lambda_k} \left[p(\mathbf{y}, \mathbf{w}|\mathbf{x}) \right] \\ &= \frac{1}{p(\mathbf{y}|\mathbf{x})} \sum_{\mathbf{w}} p(\mathbf{y}, \mathbf{w}|\mathbf{x}) \frac{\partial}{\partial \lambda_k} \left[\log p(\mathbf{y}, \mathbf{w}|\mathbf{x}) \right] \\ &= \sum_{\mathbf{w}} p(\mathbf{w}|\mathbf{y}, \mathbf{x}) \frac{\partial}{\partial \lambda_k} \left[\log p(\mathbf{y}, \mathbf{w}|\mathbf{x}) \right],\end{aligned}$$

which is the expectation of the fully-observed gradient over all the unobserved variables. Substituting the expression (2) for the fully-observed gradient yields (6).

The marginal likelihood (5) can be maximized numerically using standard techniques. In the experiments below, we use a quasi-Newton method, just as we do in the fully-observed case, but other

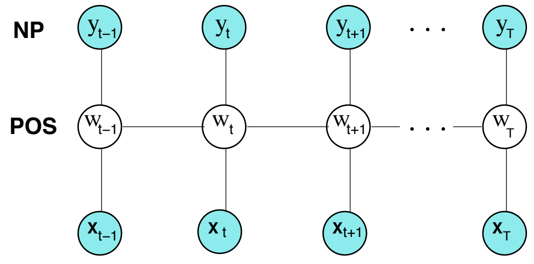


Figure 4: Graphical representation of factorial CRF for the joint noun phrase chunking and part of speech labeling problem, demonstrating the process for computing the probability $p(\mathbf{w}|\mathbf{y}^{(i)},\mathbf{x}^{(i)})$.

optimization algorithms can be used, such as expectation maximization and its variants. Unfortunately, unlike the fully-observed case, the penalized likelihood for marginal DCRFs is not convex in general. Thus standard optimization techniques are guaranteed to find only a local maximum, the quality of which depends on where in parameter space the optimizer is initialized. In Section 4.3, we evaluate an approach for finding a good starting point.

3.4.2 INFERENCE IN MARGINAL DCRFS

In this section, we discuss how to compute the model probabilities required by the marginal DCRF gradient. The solutions are very similar to those for DCRFs. The gradient in Equation (6) requires computing two kinds of marginal probabilities. First, the marginal probabilities $p(\tilde{\mathbf{y}}_{t,c},\tilde{\mathbf{w}}_{t,c}|\mathbf{x}^{(i)})$ in the second term can be computed by standard inference algorithms, just as in Section 3.2. Second, the other kind of marginal probabilities are of the form $p(\tilde{\mathbf{w}}_{t,c}|\mathbf{y}^{(i)},\mathbf{x}^{(i)})$, used in the first term on the right hand side of Equation (6). We do this by performing inference in a *clamped model*, in which $\mathbf{y}$ is fixed to its observed values (shown in Figure 4). More specifically, to form the clamped model for a training instance $\{\mathbf{x}^{(i)},\mathbf{y}^{(i)},\mathbf{w}^{(i)}\}$, we instantiate y_t nodes (nodes associated with the noun phrase labels) from $\mathbf{y}^{(i)}$. This eliminates the edges that are solely between y_t nodes, and hence $p(\mathbf{w}|\mathbf{y}^{(i)},\mathbf{x}^{(i)})$ can be computed efficiently using any inference algorithm. Finally, to compute the actual likelihood (5), we can pick an arbitrary assignment $\mathbf{w}'$ and compute the likelihood $p(\mathbf{y}|\mathbf{x}) = p(\mathbf{y},\mathbf{w}'|\mathbf{x})/p(\mathbf{w}'|\mathbf{y},\mathbf{x})$.

For decoding in marginal DCRFs, we wish to find the most likely label sequence for only the $\mathbf{y}$ variables, that is:

$$\begin{aligned}\mathbf{y}^* &= \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}) \\ &= \arg\max_{\mathbf{y}} \sum_{\mathbf{w}} p(\mathbf{y},\mathbf{w}|\mathbf{x}).\end{aligned} \tag{7}$$

To solve this maximization problem, we present an approximate algorithm that is a mixture of the max-product and sum-product BP algorithms. Basically, nodes that are marginalized out (in our example, nodes associated with the POS labels) send sum-product messages, and nodes that are not marginalized out (in our example, nodes associated with the NP labels) send max-product messages. These updates are summarized in Algorithm 2.

Algorithm 2 MAP algorithm for marginal FCRF

```
1 If u is a marginalized node, perform: $m_u(x_v) \leftarrow \sum_{x_u}\{\Phi(x_u,x_v)\prod_{x_t \neq x_v} m_t(x_u)\}$.
2 If u is not a marginalized node, perform: $m_u(x_v) \leftarrow \max_{x_u}\{\Phi(x_u,x_v)\prod_{x_t \neq x_v} m_t(x_u)\}$.
3 For all nodes perform: $p(x_u,x_v) \leftarrow \kappa\Phi(x_u,x_v)\prod_{x_t \neq x_v} m_t(x_u)\prod_{x_w \neq x_u} m_w(x_v)$.
```

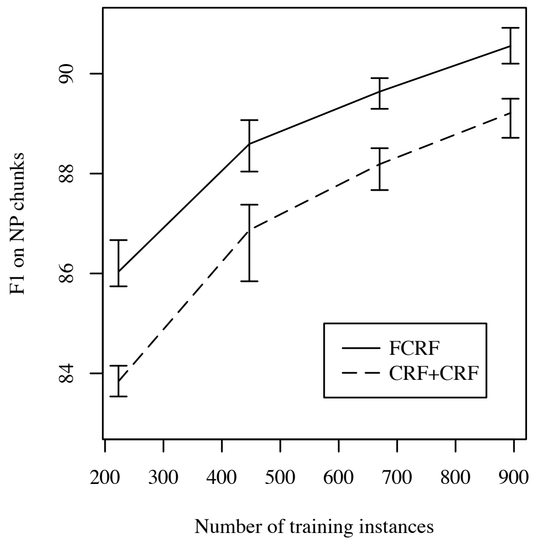


Figure 5: Performance of FCRFs and cascaded approaches on noun-phrase chunking, averaged over five repetitions. The error bars on FCRF and CRF+CRF indicate the range of the repetitions.

Finally, an important point is that part of the reason that the marginal maximization in (7) is possible in our setting is because of our choice of **w**. In the application considered here, **w** is a single chain of a two-chain FCRF, so that the marginal $p(\mathbf{y}|\mathbf{x})$ is likewise the marginal of a single chain, which can be approximated by BP in a natural way. In a more difficult case—for example, if the variables **y** are disconnected, spread throughout the model, and highly correlated—further approximation would be necessary.

4. Experiments

We present experiments comparing factorial CRFs to other approaches on noun-phrase chunking (Sang and Buchholz, 2000). Also, we compare different schedules of loopy belief propagation in factorial CRFs.

4.1 FCRFs for Noun-Phrase Chunking

Automatically finding the base noun phrases in a sentence can be viewed as a sequence labeling task by labeling each word as either BEGIN-PHRASE, INSIDE-PHRASE, or OTHER (Ramshaw and

	Size	CRF+CRF	Brill+CRF	FCRF
POS accuracy	223	86.23	N/A	**93.12**
	447	90.44		**95.43**
	670	92.33		**96.34**
	894	93.56		**96.85**
	2234	96.18		**97.87**
	8936	98.28		**98.92**
Joint accuracy	223	81.92	N/A	**89.19**
	447	86.58		**91.85**
	670	88.68		**92.86**
	894	90.06		**93.60**
	2234	93.00		**94.90**
	8936	95.56		**96.48**
NP F1	223	83.84	86.02	**86.03**
	447	86.87	88.56	**88.59**
	670	88.19	**89.65**	89.64
	894	89.21	90.31	**90.55**
	2234	91.07	91.90	**92.02**
	8936	93.10	93.33	**93.87**

Table 1: Comparison of performance of cascaded models and FCRFs on simultaneous noun-phrase chunking and POS tagging. The column Size lists the number of sentences used in training. The row CRF+CRF lists results from cascaded CRFs, and Brill+CRF lists results from a linear-chain CRF given POS tags from the Brill tagger. The FCRF always outperforms CRF+CRF, and given sufficient training data outperforms Brill+CRF. With small amounts of training data, Brill+CRF and the FCRF perform comparably, but the Brill tagger was trained on over 40,000 sentences, including some in the CoNLL 2000 test set.

Marcus, 1995). The task is typically performed by an initial pass of part-of-speech tagging, but then it can be difficult to recover from errors by the tagger. In this section, we address this problem by performing part-of-speech tagging and noun-phrase segmentation jointly in a single factorial CRF.

Our data comes from the CoNLL 2000 shared task (Sang and Buchholz, 2000), and consists of sentences from the Wall Street Journal annotated by the Penn Treebank project (Marcus et al., 1993). We consider each sentence to be a training instance, with single words as tokens. The data are divided into a standard training set of 8936 sentences and a test set of 2012 sentences. There are 45 different POS labels, and the three NP labels.[2]

We compare a factorial CRF to two cascaded approaches, which we call *CRF+CRF* and *Brill+CRF*. CRF+CRF uses one linear-chain CRF to predict POS labels, and another linear-chain CRF to predict NP labels, using as a feature the Viterbi POS labeling from the first CRF. Brill+CRF predicts NP labels using the POS labels provided from the Brill tagger, which we expect to be more accurate than those from our CRF, because the Brill tagger was trained on over four times more data, including sentences from the CoNLL 2000 test set.

The factorial CRF uses the graph structure in Figure 1(b), with one chain modeling the part-of-speech process and the other modeling the noun-phrase process. We use L-BFGS to optimize the

2. The source code used for this experiment is available at `http://mallet.cs.umass.edu/index.php/GRMM`.

$w_{t-\delta} = w$
w_t matches `[A-Z][a-z]+`
w_t matches `[A-Z]`
w_t matches `[A-Z]+`
w_t matches `[A-Z]+[a-z]+[A-Z]+[a-z]`
w_t matches `.*[0-9].*`
w_t appears in list of first names, last names, company names, days, months, or geographic entities
w_t is contained in a lexicon of words with POS T (from Brill tagger)
$T_t = T$
$q_k(\mathbf{x}, t+\delta)$ for all k and $\delta \in [-3, 3]$

Table 2: Input features $q_k(\mathbf{x},t)$ for the CoNLL data. In the above w_t is the word at position t, T_t is the POS tag at position t, w ranges over all words in the training data, and T ranges over all part-of-speech tags.

posterior $p(\Lambda|\mathcal{D})$, and TRP to compute the marginal probabilities required by $\partial\mathcal{L}/\partial\lambda_k$. Based on past experience with linear-chain CRFs, we use the prior variance $\sigma^2 = 10$ for all models.

We factorize our features as $f_k(y_{t,c}, x, t) = p_k(y_{t,c}) q_k(\mathbf{x}, t)$ where $p_k(y_{t,c})$ is a binary function on the assignment, and $q_k(\mathbf{x},t)$ is a function solely of the input string. Table 2 shows the features we use. All three approaches use the same features, with the obvious exception that the FCRF and the first stage of CRF+CRF do not use the POS features $T_t = T$.

Performance on noun-phrase chunking is summarized in Table 1. As usual, we measure performance on chunking by *precision*, the percentage of returned phrases that are correct; *recall*, the percentage of correct phrases that were returned; and their harmonic mean F_1. In addition, we also report accuracy on POS labels,[3] and joint accuracy on (POS, NP) pairs. Joint accuracy is simply the number of sequence positions for which all labels were correct.

Each row in Table 1 is the average of five different random subsets of the training data, except for row 8936, which is run on the single official CoNLL training set. All conditions used the same 2012 sentences in the official test set.

On the full training set, FCRFs perform better on NP chunking than either of the cascaded approaches, including Brill+POS. The Brill tagger (Brill, 1994) is an established part-of-speech tagger whose training set is not only over four times bigger than the CoNLL 2000 data set, but also includes the WSJ corpus from which the CoNLL 2000 test set was derived. The Brill tagger is 97% accurate on the CoNLL data. Also, note that the FCRF—which predicts both noun-phrase boundaries and POS—is more accurate than a linear-chain CRF which predicts only part of speech. Our explanation for this is that the NP chain captures long-run dependencies among the POS labels. The POS-only accuracy is listed under CRF+CRF in Table 1.

3. To simulate the effects of a cascaded architecture, the POS labels in the CoNLL-2000 training and test sets were automatically generated by the Brill tagger. Thus, POS accuracy measures agreement with the Brill tagger, not agreement with human judgements.

Method	Time (hr)		NP F1		LBFGS iter
	μ	s	μ	s	μ
Random (3)	15.67	2.90	88.57	0.54	63.6
Tree (3)	13.85	11.6	88.02	0.55	32.6
Tree (∞)	13.57	3.03	88.67	0.57	65.8
Random (∞)	13.25	1.51	88.60	0.53	76.0
Exact	20.49	1.97	88.63	0.53	73.6

Table 3: Comparison of F1 performance on the chunking task by inference algorithm. The columns labeled μ give the mean over five repetitions, and s the sample standard deviation. Approximate inference methods have labeling accuracy very similar to exact inference with lower total training time. The differences in training time between Tree (∞) and Exact and between Random (∞) and Exact are statistically significant by a paired t-test ($df = 4; p < 0.005$).

On smaller training subsets, the FCRF outperforms CRF+CRF and performs comparably to Brill+CRF. For all the training subset sizes, the difference between CRF+CRF and the FCRF is statistically significant by a two-sample t-test ($p < 0.002$). In fact, there was no subset of the data on which CRF+CRF performed better than the FCRF. The variation over the randomly selected training subsets is small—the standard deviation over the five repetitions has mean 0.39—indicating that the observed improvement is not due to chance. Performance and variance on noun-phrase chunking is shown in Figure 5.

On this data set, several systems are statistically tied for best performance. Kudo and Matsumoto (2001) report an F1 of 94.39 using a combination of voting support vector machines. Sha and Pereira (2003) give a linear-chain CRF that achieves an F1 of 94.38, using a second-order Markov assumption, and including bigram and trigram POS tags as features. An FCRF imposes a first-order Markov assumption over labels, and represents dependencies only between cotemporal POS and NP label, not POS bigrams or trigrams. Thus, Sha and Pereira's results suggest that more richly-structured DCRFs could achieve better performance than an FCRF.

Other DCRF structures can be applied to many different language tasks, including information extraction. Peshkin and Pfeffer (2003) apply a generative DBN to extraction from seminar announcements (Frietag and McCallum, 1999), attaining improved results, especially in extracting locations and speakers, by adding a factor to remember the identity of the last non-background label.

4.2 Comparison of Inference Algorithms

Because DCRFs can have rich graphical structure, and require many marginal computations during training, inference is critical to efficient training with many labels and large data sets. In this section, we compare different inference methods both on training time and labeling accuracy of the final model.

Because exact inference is feasible for a two-chain FCRF, this provides a good case to test whether the final classification accuracy suffers when approximate methods are used to calculate the gradient. Also, we can compare different methods for approximate inference with respect to speed and accuracy.

We train factorial CRFs on the noun-phrase chunking task described in the last section. We compute the gradient using exact inference and approximate belief propagation using both random and tree-based schedules, as described in Section 3.2. Algorithms are considered to have converged when no message changes by more than 10^{-3}. In these experiments, we observe that the approximate BP algorithms always converge, although this is not guaranteed in general. We train on five random subsets of 5% of the training data, and the same five subsets are used in each condition. All experiments were performed on a 2.8 GHz Intel Xeon with 4 GB of memory.

In an attempt to reduce the training time, we also examine early stopping of BP. For each message-passing schedule, we compare terminating belief propagation on convergence (Random(∞) and Tree(∞) in Table 3), to terminating after three iterations (Random (3) and Tree (3)). In all cases, we run BFGS to convergence. To be clear, training as a whole is a two-loop process: in the outer loop, BFGS updates the parameters to increase the likelihood, and in the inner loop, belief propagation passes messages to compute beliefs which are used to approximate the gradient. In these experiments, we examine early stopping in the inner loop, not the outer loop.

Thus, early-stopping of BP need not lead to faster training time overall. Of course each call the inner loop of training becomes faster with early stopping. However, early stopping of BP can interact badly with the outer loop, because it makes the gradients less accurate. If the gradient is too inaccurate, then the outer loop will require many more iterations, resulting in greater training time overall, even though the time per gradient computation is lower. Another hazard is that no maximizing step may be possible along the approximate gradient, even if one is possible along the true gradient. When that happens, the gradient descent algorithm terminates prematurely, leading to decreased performance.

Table 3 shows the average F1 score and total training times of DCRFs trained by the different inference methods. Unexpectedly, letting the belief propagation algorithms run to convergence led to lower training time than the early cutoff. For example, even though Random(3) averaged 427 sec per gradient computation compared to 571 sec for Random(∞), Random(∞) took less total time to train, because Random(∞) needed an average of 83.6 gradient computations per training run, compared to 133.2 for Random(3).

As for final classification performance, the various approximate methods and exact inference perform similarly, except that Tree(3) has lower final performance because maximization ended prematurely, averaging only 32.6 maximizer iterations. The variance in F1 over the subsets, although not large, is much larger than the F1 difference between the inference algorithms.

Previous work (Wainwright, 2002) has shown that TRP converges faster than *synchronous* belief propagation, that is, with Jacobi updates. Both the schedules discussed in Section 3.2 use asynchronous Gauss-Seidel updates. We emphasize that the graphical models in these experiments are always pairs of coupled chains. On more complicated models, or with a different choice of spanning trees, tree-based updates could outperform random asynchronous updates. Also, in complex models, the difference in classification accuracy between exact and approximate inference could be larger, but then exact inference is likely to be intractable.

In summary, we draw three conclusions about belief propagation on this particular model. First, using approximate inference instead of exact inference leads to lower overall training time with no loss in accuracy. Indeed, the two-level FCRFs that we consider here appear to have been particularly easy cases for BP, because we observed little difficulty with convergence. Second, there is little difference between a random tree schedule and a completely random schedule for belief propagation.

	Initial model	Precision	Recall	Accuracy	F1
Joint FCRF	Random	0.431	0.420	0.710	0.425
Marginal FCRF	Random	0.470	0.430	0.730	0.450
	1-Joint	0.470	0.430	0.730	0.450
	5-Joint	0.460	0.418	0.726	0.440
	10-Joint	0.460	0.418	0.730	0.440
	15-Joint	0.454	0.415	0.725	0.434
	20-Joint	0.460	0.423	0.730	0.440
	25-Joint	0.453	0.408	0.725	0.430
	Final-Joint	0.477	0.404	0.723	0.437

Table 4: NP chunking results comparing marginal FCRF and jointly trained FCRF performance using a data set consisting of 21 training instances and 195 testing instances.

Third, running belief propagation to convergence leads both to increased classification accuracy and lower overall training time than an early cutoff.

4.3 Experiments with Marginal FCRFs

In this section we apply marginal DCRFs (Section 3.4) to the CoNLL 2000 shared task data set (Sang and Buchholz, 2000), and to synthetic data.

4.3.1 NOUN-PHRASE CHUNKING

In Section 4.1 we presented experiments using FCRFs for the noun-phrase segmentation problem. In this section we apply marginal FCRFs to the same problem, where we wish to predict the noun-phrase labels and marginalize out the part-of-speech labels.

Because the marginal likelihood is not a convex function of the model parameters, our choice of initialization can affect the quality of the learned model. In order to partly capture the usefulness of the part-of-speech labels in the data, we initialize the marginal FCRF from a joint FCRF at some intermediate stages of training. We train an FCRF using the joint likelihood from a random initialization, saving the model parameters after each iteration of BFGS. Then we use each of those saved parameter settings as an initialization for marginal training; we call this n-joint initialization, where n is the number of BFGS steps on the joint likelihood used for the initializer. We compare this to initializing from a fully-trained joint FCRF (Final-Joint) and and to initializing from random parameters. We use two different-sized subsets of the CoNLL 2000 data. The first subset contains 21 training instances and 195 testing instances (Table 4). The second contains 447 training instances and 2012 testing instances (Table 5).

Based on the results shown in Tables 4 and 5, the best performance using the small data set is attained when the marginal FCRF is trained by the joint FCRF model trained for 1 iterations which improves the F1 by 0.5%. The best performance using the large data set is attained when the marginal FCRF is trained by the joint FCRF model trained for 90 iterations which improves the F1 by 0.3%. The difference between the marginal FCRF and the joint FCRF is not statistically significant, however.

	Initial model	**Precision**	**Recall**	**Accuracy**	**F1**
Joint FCRF	Random	0.819	0.803	0.913	0.811
Marginal FCRF	Random	0.814	0.800	0.910	0.806
	1-Joint	0.810	0.800	0.910	0.810
	5-Joint	0.810	0.800	0.909	0.806
	20-Joint	0.810	0.782	0.907	0.795
	30-Joint	0.820	0.791	0.908	0.804
	50-Joint	0.820	0.800	0.913	0.808
	70-Joint	0.820	0.800	0.913	0.810
	80-Joint	0.827	0.800	0.920	0.813
	90-Joint	0.827	0.800	0.920	0.814
	Final-Joint	0.825	0.797	0.913	0.811

Table 5: NP chunking results comparing marginal FCRF and jointly trained FCRF performance using a data set consisting of 447 training instances and 2012 testing instances.

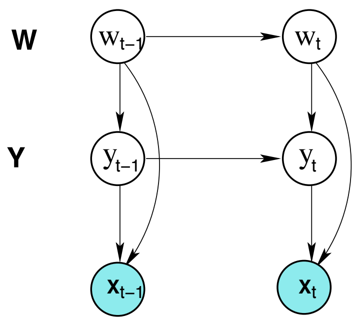


Figure 6: Graphical representation of the DBN used for generating synthetic data.

4.3.2 SYNTHETIC DATA

In this section we discuss a set of experiments with synthetic data that further highlights the differences between joint and marginal DCRFs. In this set of experiments we generate synthetic data using randomly chosen generative models that all share the graphical structure shown in Figure 6. All variables take on discrete values from a finite domain of cardinality five. The parameters (horizontal and vertical transition probability matrices, and also the observation model) are randomly selected from a uniform Dirichlet distribution with parameter $\mu = 0.5$. We then sample each model to generate a set of 200 training sequences and 400 testing sequences, each of length 20.

For each synthetic training set, we train both a joint FCRF with the graphical structure shown in Figure 3 and a marginal FCRF initialized by the final parameters of the joint FCRF. Figure 7 compares the accuracy of the marginal FCRFs to the joint FCRFs over the different training and test sets. It can be observed when the joint FCRF performs poorly, then the marginal FCRF on average performs better. In some cases the prediction accuracy of the marginal FCRF is significantly better than the prediction accuracy of the joint FCRF.

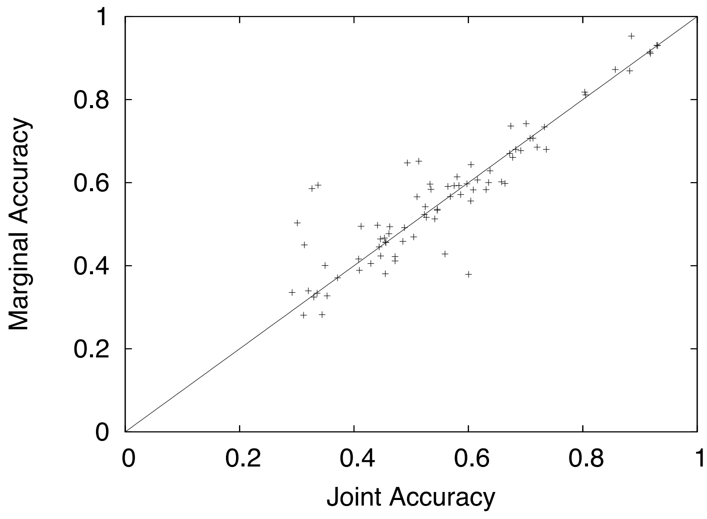


Figure 7: Accuracy of the marginal FCRF versus accuracy of the joint FCRF over the factor $\mathbf{Y}$. Each point represents a training and testing set generated from a different randomly-selected generative model with structure shown in Figure 6.

If the joint FCRF performs well, however, the marginal FCRF tends to yield the same prediction accuracy. We conjecture that when we have learned a good joint FCRF model given a set of training sequences, the marginally trained FCRF initialized by the joint FCRF does not offer improvement in accuracy. However, when the joint FCRF performs poorly, we can train a marginal FCRF that is likely to outperform the joint FCRF.

In order to further study this phenomenon, we measure the entropy of the model distribution, the idea being that when the model is very accurate, it has little uncertainty over the latent variables, so modeling the marginal directly is unlikely to make a difference. To measure the uncertainty over output labels, we use a *per-timestep entropy* measure, that is, $\sum_i \sum_t H(p(y_t|\mathbf{x}^{(i)}))$, where as before i ranges over test instances, and t over sequence positions. Figure 8 shows the plot of the accuracy of the joint FCRF versus its per-timestep entropy. As the per-timestep entropy of the model increases, the accuracy decreases. This suggests that the per-timestep entropy can serve as a surrogate measure to decide which problems are most appropriate for marginal training.

Figure 9 plots the ratio of the the marginal FCRF accuracy to the joint FCRF accuracy, as a function of the per-timestep entropy of the joint FCRF. We observe that for joint FCRF models with a smaller entropy measure, this ratio is close to one which means that both joint FCRF and marginal FCRF perform almost the same. However, for joint FCRF models with high entropy, this ratio increases on average, meaning that the marginal FCRF is outperforming the joint FCRF. This suggests that the per-timestep entropy of the jointly trained model provides some indication of whether marginal training may be expected to improve performance.

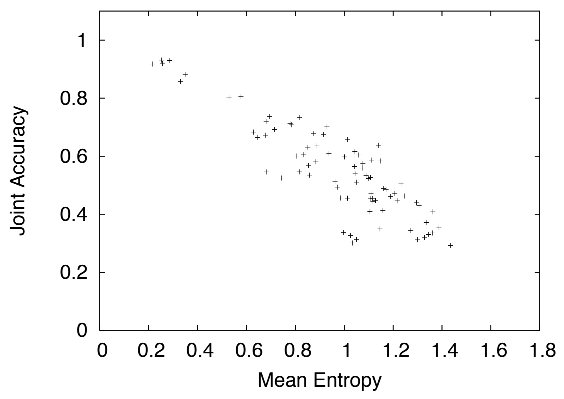


Figure 8: Accuracy of the joint FCRF as a function of the mean per-timestep entropy, $E(H(p(y_t|\mathbf{x})))$, averaged over all time steps $\mathbf{t}$, and all testing sequences $\mathbf{x}^{(i)}$, of the joint FCRF.

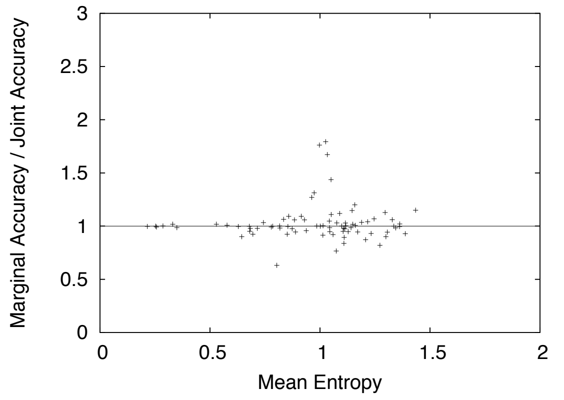


Figure 9: Ratio of the accuracy of the marginal FCRF accuracy to the the joint FCRF accuracy, as a function of the mean per-timestep entropy of the joint FCRF.

4.4 Cascaded Training for Transfer Learning

In this section, we consider an application of DCRFs to transfer learning, both as an additional application of DCRFs, and as an evaluation of the cascaded training procedure described in Section 3.3.3. The task is to extract the details of an academic seminar—including its starting time, ending time, location, and speaker—from an email announcement. The data is a collection of 485 e-mail messages announcing seminars at Carnegie Mellon University, gathered by Freitag (1998), and has been the subject of much previous work using a wide variety of learning methods. Despite all this work, however, the best reported systems have precision and recall on speaker names and locations of only about 75%—too low to use in a practical system. This task is so challenging because

$w_t = w$
w_t matches `[A-Z][a-z]+`
w_t matches `[A-Z][A-Z]+`
w_t matches `[A-Z]`
w_t matches `[A-Z]+`
w_t matches `[A-Z]+[a-z]+[A-Z]+[a-z]`
w_t appears in list of first names, last names, honorifics, etc.
w_t appears to be part of a time followed by a dash
w_t appears to be part of a time preceded by a dash
w_t appears to be part of a date
$T_t = T$
$q_k(\mathbf{x}, t+\delta)$ for all k and $\delta \in [-4,4]$

Table 6: Input features $q_k(\mathbf{x},t)$ for the seminars data. In the above w_t is the word at position t, T_t is the POS tag at position t, w ranges over all words in the training data, and T ranges over all Penn Treebank part-of-speech tags. The “appears to be” features are based on hand-designed regular expressions that can span several tokens.

System		stime	etime	location	speaker	overall
WHISK	Soderland (1999)	92.6	86.1	66.6	18.3	65.9
SRV	Freitag (1998)	98.5	77.9	72.7	56.3	76.4
HMM	Frietag and McCallum (1999)	98.5	62.1	78.6	76.6	78.9
RAPIER	Califf and Mooney (1999)	95.9	94.6	73.4	53.1	79.3
SNOW-IE	Roth and Wen-tau Yih (2001)	**99.6**	96.3	75.2	73.8	86.2
$(\mathrm{LP})^2$	Ciravegna (2001)	99.0	95.5	75.0	**77.6**	86.8
CRF (no transfer)	This paper	99.1	**97.3**	81.0	73.7	87.8
FCRF (cascaded)	This paper	99.2	96.0	84.3	74.2	88.4
FCRF (joint)	This paper	99.1	96.0	**85.3**	76.3	**89.2**

Table 7: Comparison of F_1 performance on the seminars data. Joint decoding performs significantly better than cascaded decoding. The overall column is the mean of the other four. (This table was adapted from Peshkin and Pfeffer (2003).)

the messages are written by many different people, who each have different ways of presenting the announcement information.

Because the task includes finding locations and person names, the output of a named-entity tagger is a useful feature. It is not a perfectly indicative feature, however, because many other kinds of person names appear in seminar announcements—for example, names of faculty hosts, departmental secretaries, and sponsors of lecture series. For example, the token *Host:* indicates strongly that what follows is a person name, but that person is not the seminar's speaker.

Even so, named-entity predictions do improve performance on this task. Therefore, we wish to do transfer learning from the named-entity task to the seminar announcement task. We do not have data that is labeled for both named-entity and seminar fields, so we use the cascaded training

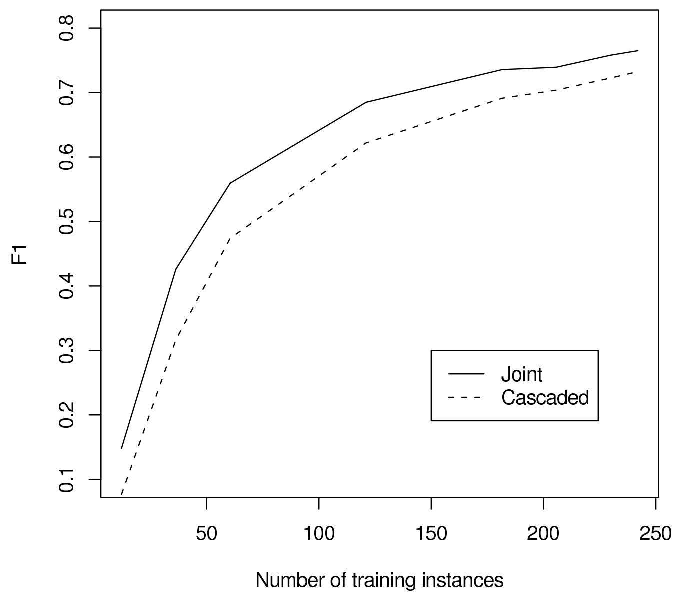


Figure 10: Learning curves for the seminars data set on the speaker field, averaged over 10-fold cross validation. Joint training performs equivalently to cascaded decoding with 25% more data.

procedure in Section 3.3.3. We are interested in two comparisons: (a) between the FCRF trained to incorporate transfer and a comparable linear-chain CRF, and (b) at test time, between cascaded decoding or joint decoding. By *cascaded decoding*, we mean an analogous procedure to cascaded training, in which the maximum-value assignment to the first level of the DCRF is computed without reference to the second level, then this assignment is clamped and decoding proceeds in the second level only. By *joint decoding*, we mean standard max-product inference in the full FCRF. We might expect joint decoding to perform better because of helpful feedback between the tasks: Information from the seminar-field predictions can improve named-entity predictions, which in turn improve the seminar-field predictions.

We use the predictions from a CRF named-entity tagger that we train on the standard CoNLL 2003 English data set. The CoNLL 2003 data set consists of newswire articles from Reuters labeled as either people, locations, organizations, or miscellaneous entities. It is much larger than the seminar announcements data set. While the named-entity data contains 203,621 tokens for training, the seminar announcements data set contains only slightly over 60,000 training tokens.

Previous work on the seminars data has used a one-field-per-document evaluation. That is, for each field, the CRF selects a single field value from its Viterbi path, and this extraction is counted as correct if it exactly matches any of the true field mentions in the document. We compute precision and recall following this convention, and report their harmonic mean F_1. As in the previous work, we use 10-fold cross validation with a 50/50 training/test split. We use a spherical Gaussian prior on parameters with variance $\sigma^2 = 0.5$.

We evaluate whether joint decoding with cascaded training performs better than cascaded training and decoding. Table 7 compares cascaded and joint decoding for CRFs with other previous results from the literature.[4] The features we use are listed in Table 6. Although previous work has

4. We omit one relevant paper (Peshkin and Pfeffer, 2003) because its evaluation method differs from all the other previous work.

used very different feature sets from ours, all of our models use exactly the same features, including the no-transfer CRF baseline.

On the most challenging fields, location and speaker, cascaded transfer is more accurate than no transfer at all, and joint decoding is more accurate than cascaded decoding. In particular, for speaker, we see an error reduction of 8% by using joint decoding over cascaded. The difference in F1 between cascaded and joint decoding is statistically significant for speaker (paired *t*-test; $p = 0.017$) but only marginally significant for location ($p = 0.067$). Our results are competitive with previous work; for example, on location, the CRF is more accurate than any of the existing systems, and the CRF has the highest overall performance, that is, averaged over all fields, than the previously reported systems.

Figure 10 shows the difference in performance between joint and cascaded decoding as a function of training set size. Cascaded decoding with the full training set of 242 emails performs equivalently to joint decoding on only 181 training instances, a 25% reduction in the training set.

Examining the trained models, we can observe errors made by the general-purpose named entity tagger, and how they can be corrected by considering the seminars labels. In newswire text, long runs of capitalized words are rare, often indicating the name of an entity. In email announcements, runs of capitalized words are common in formatted text blocks like:

```
Location:  Baker Hall
    Host:  Michael Erdmann
```

In this type of situation, the general named entity tagger often mistakes *Host:* for the name of an entity, especially because the word preceding *Host* is also capitalized. On one of the cross-validated testing sets, of 80 occurrences of the word *Host:*, the named-entity tagger labels 52 as some kind of entity. When joint decoding is used, however, only 20 occurrences are labeled as entities. Recall that in both of these settings, training is performed in exactly the same way; the only difference is that joint decoding takes into account information about the seminar labels when choosing named-entity labels. This is an example of how domain-specific information from the main task can improve performance on a more standard, general-purpose subtask.

5. Related Work

Since the original work on conditional random fields (Lafferty et al., 2001), there has been much interest in training discriminative models with more general graphical structures. One of the first such applications was relational Markov networks (Taskar et al., 2002), which were first applied to collective classification of Web pages. There has also been interest in grid-structured loopy CRFs for computer vision (He et al., 2004; Kumar and Hebert, 2003), in which jointly-trained Markov random fields are a classical technique. Another type of structured problem which has seen some attention in the literature is discriminative learning of distributions over context-free parse trees, in which training has done done using max-margin methods (Taskar et al., 2004b; McDonald et al., 2005) and perceptron-like methods (Viola and Narasimhan, 2005).

The marginal DCRF is an example of a CRF with latent variables, a model class that has received some recent attention. Recent examples of latent variable CRFs include Quattoni et al. (2005), in which the latent variables label parts of an object in an image, and McCallum et al. (2005), in which the latent structure is an alignment of two sequences. The training techniques described here here can be applied more generally to latent-variable CRFs. Alternatively, latent-variable CRFs can be trained using EM (McCallum et al., 2005), which is described in general in Sutton and McCallum (2006). Latent-variable CRFs are closely related to neural networks, and many training techniques from that literature can be applied here.

Currently, the most popular alternative approaches to training structured discriminative models are maximum-margin training (Taskar et al., 2004a; Altun et al., 2003), and perceptron training (Collins, 2002), which has been especially popular in NLP because of its ease of implementation.

The factorial CRF that we present here should not be confused with the factorial Markov random fields that have been proposed in the computer vision community (Kim and Zabih, 2002). In that model, each of the factors is a grid, rather than a chain, and they interact through a directed model, as in a factorial HMM.

The DCRF application to transfer learning in Section 4.4 is reminiscent of stacking (Wolpert, 1992). The most notable difference is that because the levels are decoded jointly, information from later levels can affect the decisions made about earlier ones.

Finally, some results presented here have appeared in earlier conference versions, in particular the results on noun-phrase chunking (Sutton et al., 2004) and transfer learning (Sutton and McCallum, 2005).

6. Conclusions

Dynamic CRFs are conditionally-trained undirected sequence models with repeated graphical structure and tied parameters. They combine the best of both conditional random fields and the widely successful dynamic Bayesian networks (DBNs). DCRFs address difficulties both of DBNs, by easily incorporating arbitrary overlapping input features, and of previous conditional models, by allowing more complex dependence between labels. Inference in DCRFs can be done using approximate methods, and training can be done by maximum a posteriori estimation.

Empirically, we have shown that factorial CRFs can be used to jointly perform several labeling tasks at once, sharing information between them. Such a joint model performs better than a model that does the individual labeling tasks sequentially, and has potentially many practical implications, because cascaded models are ubiquitous in NLP. Also, we have shown that using approximate inference leads to lower total training time with no loss in accuracy.

In future research, we plan to explore other inference methods to make training more efficient, including expectation propagation (Minka, 2001b), contrastive divergence (Hinton, 2002) and variational approximations. Finally, investigating other DCRF structures, such as hierarchical CRFs and DCRFs with memory of previous labels, could lead to applications into many of the tasks to which DBNs have been applied, including object recognition, speech processing, and bioinformatics.

Acknowledgments

We thank three anonymous reviewers for many helpful comments on an earlier version of this work, and we thank Kevin Murphy for helpful conversations. This work was supported in part by the Center for Intelligent Information Retrieval; by SPAWARSYSCEN-SD grant number N66001-02-1-8903; by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under contract number NBCHD030010; by the Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grants #IIS-0427594 and #IIS-0326249; and in part by the Defense Advanced Research Projects Agency (DARPA) under contract number HR0011-06-C-0023. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors.

References

Srinivas M. Aji, Gavin B. Horn, and Robert J. McEliece. On the convergence of iterative decoding on graphs with a single cycle. In *Proc. IEEE Int'l Symposium on Information Theory*, 1998.

Yasemin Altun, Ioannis Tsochantaridis, and Thomas Hofmann. Hidden Markov support vector machines. In *International Conference on Machine Learning (ICML)*, 2003.

Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

Jeff Bilmes. Graphical models and automatic speech recognition. In M. Johnson, S.P. Khudanpur, M. Ostendorf, and R. Rosenfeld, editors, *Mathematical Foundations of Speech and Language Processing*. Springer-Verlag, 2003.

Eric Brill. Some advances in rule-based part of speech tagging. In *National Conference on Artificial Intelligence (AAAI)*, 1994.

Hung H. Bui, Svetha Venkatesh, and Geoff West. Policy recognition in the Abstract Hidden Markov Model. *Journal of Artificial Intelligence Research*, 17, 2002.

Mary Elaine Califf and Raymond J. Mooney. Relational learning of pattern-match rules for information extraction. In *National Conference on Artificial Intelligence (AAAI)*, pages 328–334, 1999.

Fabio Ciravegna. Adaptive information extraction from text by rule induction and generalisation. In *International Joint Conference on Artificial Intelligence (ICML)*, 2001.

Michael Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2002.

Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3):142–150, 1989.

Gal Elidan, Ian McGraw, and Daphne Koller. Residual belief propagation: Informed scheduling for asynchronous message passing. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2006.

Shai Fine, Yoram Singer, and Naftali Tishby. The hierarchical hidden Markov model: Analysis and applications. *Machine Learning*, 32(1):41–62, 1998.

Dayne Freitag. *Machine Learning for Information Extraction in Informal Domains*. PhD thesis, Carnegie Mellon University, 1998.

Dayne Frietag and Andrew McCallum. Information extraction with HMMs and shrinkage. In *AAAI Workshop on Machine Learning for Information Extraction*, 1999.

Zoubin Ghahramani and Michael I. Jordan. Factorial hidden Markov models. *Machine Learning*, (29):245–273, 1997.

Xuming He, Richard S. Zemel, and Miguel Á. Carreira-Perpiñián. Multiscale conditional random fields for image labelling. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004.

Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:1771–1800, 2002.

Junhwan Kim and Ramin Zabih. Factorial Markov random fields. In *European Conference on Computer Vision (ECCV)*, pages 321–334, 2002.

Taku Kudo and Yuji Matsumoto. Chunking with support vector machines. In *Conference of the North American Chapter of the Association for Computation Linguistics (NAACL)*, 2001.

Sanjiv Kumar and Martial Hebert. Discriminative fields for modeling spatial dependencies in natural images. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems (NIPS) 16*. MIT Press, Cambridge, MA, 2003.

John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *International Conference on Machine Learning (ICML)*, 2001.

Robert Malouf. A comparison of algorithms for maximum entropy parameter estimation. In Dan Roth and Antal van den Bosch, editors, *Conference on Natural Language Learning (CoNLL)*, pages 49–55, 2002.

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, MA, 1999.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

Andrew McCallum and David Jensen. A note on the unification of information extraction and data mining using conditional-probability, relational models. In *IJCAI'03 Workshop on Learning Statistical Models from Relational Data*, 2003.

Andrew McCallum, Dayne Freitag, and Fernando Pereira. Maximum entropy Markov models for information extraction and segmentation. In *International Conference on Machine Learning (ICML)*, pages 591–598. Morgan Kaufmann, San Francisco, CA, 2000.

Andrew McCallum, Kedar Bellare, and Fernando Pereira. A conditional random field for discriminatively-trained finite-state string edit distance. In *Conference on Uncertainty in AI (UAI)*, 2005.

Ryan McDonald, Koby Crammer, and Fernando Pereira. Online large-margin training of dependency parsers. In *Proceedings of the Annual Meeting of the ACL*, pages 91–98, 2005.

Thomas P. Minka. The EP energy function and minimization schemes. `http://research.microsoft.com/~minka/papers/ep/minka-ep-energy.pdf`, 2001a.

Tom Minka. Divergence measures and message passing. Technical Report MSR-TR-2005-173, Microsoft Research, 2005.

Tom Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, MIT, 2001b.

Mehryar Mohri, Fernando Pereira, and Michael Riley. Weighted finite-state transducers in speech recognition. *Computer Speech and Language*, 16(1):69–88, 2002.

Kevin Murphy and Mark A. Paskin. Linear time inference in hierarchical HMMs. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

Kevin P. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, U.C. Berkeley, July 2002.

Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 467–475, 1999.

Ara V. Nefian, Luhong Liang, Xiaobo Pi, Liu Xiaoxiang, Crusoe Mao, and Kevin Murphy. A coupled HMM for audio-visual speech recognition. In *IEEE Int'l Conference on Acoustics, Speech and Signal Processing*, pages 2013–2016, 2002.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer-Verlag, New York, 1999. ISBN 0-387-98793-2.

Leonid Peshkin and Avi Pfeffer. Bayesian information extraction network. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

David Pinto, Andrew McCallum, Xing Wei, and W. Bruce Croft. Table extraction using conditional random fields. In *Proceedings of the ACM SIGIR*, 2003.

Ariadna Quattoni, Michael Collins, and Trevor Darrell. Conditional random fields for object recognition. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1097–1104. MIT Press, Cambridge, MA, 2005.

Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257 – 286, 1989.

Lance A. Ramshaw and Mitchell P. Marcus. Text chunking using transformation-based learning. In *Proceedings of the Third ACL Workshop on Very Large Corpora*, 1995.

Adwait Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Conference on Empirical Methods in Natural Language Proceeding (EMNLP)*, 1996.

Dan Roth and Wen-tau Yih. Relational learning via propositional algorithms: An information extraction case study. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1257–1263, 2001.

Erik F. Tjong Kim Sang and Sabine Buchholz. Introduction to the CoNLL-2000 shared task: Chunking. In *Proceedings of CoNLL-2000 and LLL-2000*, 2000. See `http://lcg-www.uia.ac.be/~erikt/research/np-chunking.html`.

Fei Sha and Fernando Pereira. Shallow parsing with conditional random fields. In *Conference on Human Language Technology and North American Association for Computational Linguistics (HLT-NAACL)*, pages 213–220, 2003.

Marios Skounakis, Mark Craven, and Soumya Ray. Hierarchical hidden Markov models for information extraction. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, 2003.

Stephen Soderland. Learning information extraction rules for semi-structured and free text. *Machine Learning*, pages 233–272, 1999.

Charles Sutton and Andrew McCallum. Composition of conditional random fields for transfer learning. In *Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT-EMNLP)*, 2005.

Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2006. To appear.

Charles Sutton, Khashayar Rohanimanesh, and Andrew McCallum. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. In *International Conference on Machine Learning (ICML)*, 2004.

Ben Taskar, Pieter Abbeel, and Daphne Koller. Discriminative probabilistic models for relational data. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2002.

Ben Taskar, Carlos Guestrin, and Daphne Koller. Max-margin Markov networks. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004a.

Ben Taskar, Dan Klein, Michael Collins, Daphne Koller, and Chris Manning. Max-margin parsing. In *Empirical Methods in Natural Language Processing (EMNLP04)*, 2004b.

Georgios Theocharous, Khashayar Rohanimanesh, and Sridhar Mahadevan. Learning hierarchical partially observable Markov decision processes for robot navigation. In *Proceedings of the IEEE Conference on Robotics and Automation*, 2001.

Paul Viola and Mukund Narasimhan. Learning to extract information from semi-structured text using a discriminative context free grammar. In *Proceedings of the ACM SIGIR*, 2005.

S.V.N. Vishwanathan, Nicol N. Schraudolph, Mark W. Schmidt, and Kevin Murphy. Accelerated training of conditional random fields with stochastic meta-descent. In *International Conference on Machine Learning (ICML)*, pages 969–976, 2006.

Martin Wainwright. *Stochastic processes on graphs with cycles: geometric and variational approaches*. PhD thesis, MIT, 2002.

Martin Wainwright, Tommi Jaakkola, and Alan S. Willsky. Tree-based reparameterization for approximate estimation on graphs with cycles. *Advances in Neural Information Processing Systems (NIPS)*, 2001.

Hanna Wallach. Efficient training of conditional random fields. M.Sc. thesis, University of Edinburgh, 2002.

David H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.

Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51(7): 2282–2312, July 2005.

Journal of Machine Learning Research 8 (2007) 725-760 Submitted 9/05; Revised 1/07; Published 4/07

The Pyramid Match Kernel: Efficient Learning with Sets of Features

Kristen Grauman GRAUMAN@CS.UTEXAS.EDU
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712, USA

Trevor Darrell TREVOR@CSAIL.MIT.EDU
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Editor: Pietro Perona

Abstract

In numerous domains it is useful to represent a single example by the set of the local features or parts that comprise it. However, this representation poses a challenge to many conventional machine learning techniques, since sets may vary in cardinality and elements lack a meaningful ordering. Kernel methods can learn complex functions, but a kernel over unordered set inputs must somehow solve for correspondences—generally a computationally expensive task that becomes impractical for large set sizes. We present a new fast kernel function called the *pyramid match* that measures partial match similarity in time linear in the number of features. The pyramid match maps unordered feature sets to multi-resolution histograms and computes a weighted histogram intersection in order to find implicit correspondences based on the finest resolution histogram cell where a matched pair first appears. We show the pyramid match yields a Mercer kernel, and we prove bounds on its error relative to the optimal partial matching cost. We demonstrate our algorithm on both classification and regression tasks, including object recognition, 3-D human pose inference, and time of publication estimation for documents, and we show that the proposed method is accurate and significantly more efficient than current approaches.

Keywords: kernel, sets of features, histogram intersection, multi-resolution histogram pyramid, approximate matching, object recognition

1. Introduction

In a variety of domains, it is often natural and meaningful to represent a data object with a collection of its parts or component features. For instance, in computer vision, an image may be described by local features extracted from patches around salient interest points, or a shape may be described by local descriptors defined at edge pixels. Likewise, in natural language processing, documents or topics may be represented by sets or bags of words; in computational biology, a disease may be characterized by sets of gene-expression data from multiple patients. In such cases, one set of feature vectors denotes a single instance of a particular class of interest (an object, shape, document, etc.). The number of features per example varies, and within a single instance the component features may have no inherent ordering.

Classification and regression with these sets (or bags) of features is challenging. Kernel-based learning methods are appealing for their generalization ability and efficiency, however conventional

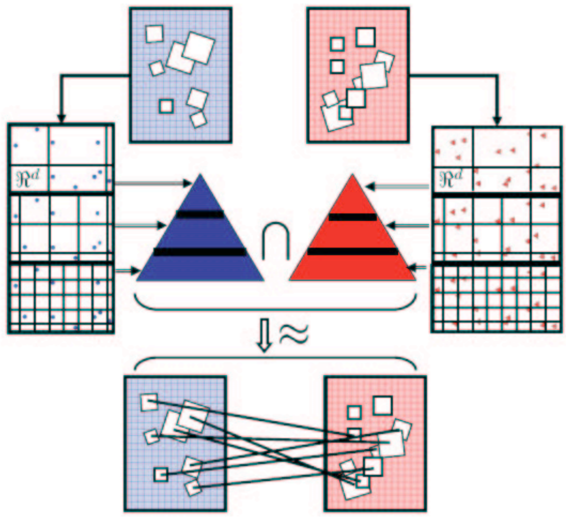


Figure 1: The pyramid match intersects histogram pyramids formed over sets of features, approximating the optimal correspondences between the sets' features. For example, vectors describing the appearance or shape within local image patches can be used to form a feature set for each image; the pyramid match approximates the similarity according to a partial matching in that feature space. (The feature space can be any local description of a data object, images or otherwise; for images the features typically will *not* be the spatial image coordinates.)

kernels are designed to operate on fixed-length vector inputs, where each dimension corresponds to a particular global attribute for that instance; the commonly used general-purpose kernels defined on $\Re^n$ inputs are not applicable in the space of vector sets. Existing kernel-based approaches specially designed for matching sets of features generally require either solving for explicit correspondences between features (which is computationally costly and prohibits the use of large inputs) or fitting a particular parametric distribution to each set (which makes restrictive assumptions about the data and can also be computationally expensive).

In this work we present the *pyramid match kernel*—a new kernel function over unordered feature sets that allows them to be used effectively and efficiently in kernel-based learning methods. Each feature set is mapped to a multi-resolution histogram that preserves the individual features' distinctness at the finest level. The histogram pyramids are then compared using a weighted histogram intersection computation, which we show defines an implicit correspondence based on the finest resolution histogram cell where a matched pair first appears (see Figure 1).

The similarity measured by the pyramid match approximates the similarity measured by the optimal correspondences between feature sets of unequal cardinality (i.e., the *partial matching* that optimally maps points in the lower cardinality set to some subset of the points in the larger set, such that the sum of the distances between matched points is minimized). Our kernel is extremely efficient and can be computed in time that is linear in the sets' cardinality. We show that the kernel function is positive-definite, meaning that it is appropriate to use with learning methods that guarantee convergence to a unique optimum only for positive-definite kernels (e.g., the support vector machine). We also provide theoretical approximation bounds for the pyramid match cost relative to the optimal partial matching cost.

Because it does not penalize the presence of superfluous data points, the proposed kernel is robust to clutter. As we will show, this translates into the ability to handle common issues faced in vision tasks like object recognition or pose estimation: unsegmented images, poor segmentations, varying backgrounds, and occlusions. The kernel also respects the co-occurrence relations inherent in the input sets: rather than matching features in a set individually, ignoring potential dependencies conveyed by features within one set, our similarity measure captures the features' joint statistics.

Other approaches to this problem have recently been proposed (Wallraven et al. 2003; Lyu 2005; Boughhorbel et al. 2004; Kondor and Jebara 2003; Wolf and Shashua 2003; Moreno et al. 2003; Shashua and Hazan 2005; Cuturi and Vert 2005; Boughorbel et al. 2005; Lafferty and Lebanon 2002), but unfortunately each suffers from some number of the following drawbacks: computational complexities that make large feature set sizes infeasible; limitations to parametric distributions which may not adequately describe the data; kernels that are not positive-definite; limitations to sets of equal size; and failure to account for dependencies within feature sets.

Our method addresses each of these issues, resulting in a kernel appropriate for comparing unordered, variable-sized feature sets within any existing kernel-based learning paradigm. We demonstrate our algorithm in a variety of classification and regression tasks: object recognition from sets of image patch features, 3-D human pose inference from sets of local contour features from monocular silhouettes, and documents' time of publication estimation from bags of local latent semantic features. The results show that the proposed approach achieves an accuracy that is comparable to or better than that of state-of-the-art techniques, while requiring significantly less computation time.

2. Related Work

In this section, we review relevant work on learning with sets of features, using kernels and support vector machines (SVMs) for recognition, and multi-resolution image representations.

Kernel-based learning algorithms, which include SVMs, kernel PCA, and Gaussian Processes, have become well-established tools that are useful in a variety of contexts, including discriminative classification, regression, density estimation, and clustering (Shawe-Taylor and Cristianini, 2004; Vapnik, 1998; Rasmussen and Williams, 2006). However, conventional kernels (such as the Gaussian RBF or polynomial) are designed to operate on $\Re^n$ vector inputs, where each vector entry corresponds to a particular global attribute for that instance. As a result, initial approaches using SVMs for recognition were forced to rely on global image features—ordered features of equal length measured from the image as a whole, such as color or grayscale histograms or vectors of raw pixel data (Chapelle et al., 1999; Roobaert and Hulle, 1999; Odone et al., 2005). Such global representations are known to be sensitive to real-world imaging conditions, such as occlusions, pose changes, or image noise.

Recent work has shown that local features invariant to common image transformations (e.g., SIFT, Lowe, 2004) are a powerful representation for recognition, because the features can be reliably detected and matched across instances of the same object or scene under different viewpoints, poses, or lighting conditions. Most approaches, however, perform recognition with local feature representations using nearest-neighbor (e.g., Belongie et al., 2002; Grauman and Darrell, 2004; Sivic and Zisserman, 2003; Berg et al., 2005) or voting-based classifiers followed by an alignment step (e.g., Lowe, 2004; Mikolajczyk and Schmid, 2001); both may be impractical for large training sets, since their classification times increase with the number of training examples. A support vector classifier or regressor, on the other hand, identifies a sparse subset of the training examples (the support vectors) to delineate a decision boundary or approximate a function of interest.

In order to more fully leverage existing kernel-based learning tools for situations where the data cannot be naturally represented by a Euclidean vector space—such as graphs, strings, or trees—researchers have developed specialized similarity measures (Gartner, 2003). In fact, due to the increasing prevalence of data that is best represented by sets of local features, several researchers have recently designed kernel functions that can handle unordered sets as input (Lyu 2005; Kondor and Jebara 2003; Wolf and Shashua 2003; Shashua and Hazan 2005; Boughhorbel et al. 2004; Boughorbel et al. 2005; Wallraven et al. 2003; Cuturi and Vert 2005; Moreno et al. 2003; Lafferty and Lebanon 2002). Nonetheless, current approaches are either prohibitively computationally expensive, are forced to make assumptions regarding the parametric form of the features, discard information by replacing inputs with prototypical features, ignore important co-occurrence information by considering features independently, are not positive-definite, and (or) are limited to sets of equal size. In addition, none have shown the ability to learn a real-valued function from sets of features; results have only been shown for classification tasks. See Figure 2 for a concise comparison of the approaches.

Approaches which fit a parametric model to feature sets in order to compare their distributions (Kondor and Jebara, 2003; Moreno et al., 2003; Cuturi and Vert, 2005; Lafferty and Lebanon, 2002) can be computationally costly and have limited applicability, since they assume both that features within a set will conform to the chosen distribution, and that sets will be adequately large enough to extract an accurate estimate of the distribution's parameters. These assumptions are violated regularly by real data, which will often exhibit complex variations within a single bag of features (e.g., patches from an image), and will produce wide ranges of cardinalities per instance (e.g., titles of documents have just a few word features). Our method instead takes a non-parametric, "model-free" approach, representing sets of features directly with multi-dimensional, multi-resolution histograms.

Kernel methods that use explicit correspondences between two sets' features search one set for the best matching feature for each member in the other, and then define set similarity as a function over those component similarity values (Wallraven et al., 2003; Lyu, 2005; Boughhorbel et al., 2004; Boughorbel et al., 2005). These methods have complexities that are quadratic in the number of features, hindering usage for kernel-based learning when feature sets are large. The "intermediate" matching kernel of Boughorbel et al. (2005) has a quadratic run-time if the number of prototypes $p = O(m)$. That is reduced if p is set so that $p < m$; however, the authors note that higher values of p yield more accurate results. Furthermore, matching each input feature independently ignores useful information about intra-set dependencies. In contrast, our kernel captures the joint statistics of co-occurring features by matching them concurrently as a set.

In the method of Wolf and Shashua (2003), similarity is measured in terms of the principal angle between the linear subspaces spanned by two sets' vector elements; the kernel has a cubic complexity and is only positive-definite for sets of equal cardinality. In the work of Shashua and Hazan (2005), an algebraic kernel is used to combine similarities given by local (vector-based) kernels, with the weighting chosen to reflect whether the features are in alignment (ordered). When set cardinalities vary, inputs are padded with zeros so as to form equal-size matrices; results are only shown for a classification task with input sets whose features' ordering is known.

Method	Complexity	Captures co-occurrences	Positive-definite	Non-parametric	Handles unequal cardinalities
Match	$O(dm^2)$			x	x
Exponent match	$O(dm^2)$		x	x	x
Greedy match	$O(dm^2)$	x		x	x
Principal angles	$O(dm^3)$	x	x		
Intermediate	$O(dpm)$		x	x	x
Bhattacharyya's	$O(dm^3)$	x	x		x
KL-divergence	$O(dm^2)$	x			x
Pyramid match	$O(dm \log D)$	x	x	x	x

Figure 2: Comparing the properties of kernel approaches to matching unordered sets. "Match" refers to the kernel of Wallraven et al. (2003), "Exponent match" is the kernel of Lyu (2005), "Greedy match" is from Boughhorbel et al. (2004), "Principal angles" is from Wolf and Shashua (2003), "Intermediate" is from Boughorbel et al. (2005), "Bhattacharyya's" is from Kondor and Jebara (2003), and "KL-divergence" is from Moreno et al. (2003). "Pyramid match" refers to the proposed kernel. Each method's computational cost is for computing a single kernel value. d is vector dimension, m is maximum set cardinality, p is the number of prototype features used by Boughorbel et al. (2005), and D is the value of the maximal feature range.

Several computer vision researchers have transformed the set of real-valued feature vectors coming from one image into a single flat histogram that counts the frequency of occurrence of some number of pre-defined (quantized) feature prototypes. In this way the quantized feature space provides a *visual vocabulary* or *bag-of-words* vector representation which can be used in conjunction with some vector-based kernels or similarity measures. This type of representation was explored for texture recognition using nearest-neighbors (Leung and Malik, 2001; Hayman et al., 2004), and more recently has been shown for classification of object categories using SVMs, Naïve Bayes classifiers, and a probabilistic Latent Semantic Analysis framework (Csurka et al., 2004; Willamowski et al., 2004; Sivic et al., 2005).

The bag-of-words is appealing because it allows existing vector-based methods to be applied and can describe the overall distribution of features in an image. However, this representation faces the substantial challenge of generating an appropriately descriptive quantization of the feature space. Bin boundary issues have been shown to create matching problems for flat histograms (Rubner et al., 2000), and though the right size of a vocabulary for a given data set can be critical to recognition performance (Csurka et al., 2004; Grauman and Darrell, 2004) it must still be determined empirically. Generating the vocabulary from large amounts of data is generally computationally costly, and it is not clear whether a generic or universal feature quantization is more or less effective than data set-dependent vocabularies. Finally, unlike the approach we develop here, existing bag-of-words techniques can only compare entire images to one another, do not allow partial matchings, and cannot be used to extract correspondences telling which features match to which. In our experiments we find that the pyramid match outperforms the bag-of-words approach for object category recognition on a challenging data set (see Section 9).

An alternative approach to discriminative classification when dealing with unordered set data is to designate prototypical examples from each class, and then represent examples by a vector giving their distances to each prototype; standard algorithms that handle vectors in a Euclidean space are then applicable. Zhang and Malik (2003) build such a classifier for handwritten digits, and use the shape context distance of Belongie et al. (2002) as the measure of similarity. Shan et al. also explore a representation based on distances to prototypes in order to avoid feature matching when recognizing vehicles (Shan et al., 2005). The issues faced by such a prototype-based method are determining which examples should serve as prototypes, choosing how many there should be, and updating the prototypes properly when new types of data are encountered. The method of Holub et al. (2005a) uses a hybrid generative-discriminative approach for object recognition, combining the Fisher kernel (Jaakkola and Haussler, 1999) and a probabilistic constellation model.

Our feature representation is based on a multi-resolution histogram, or pyramid, which is computed by binning data points into discrete regions of increasingly larger size. Single-level histograms have been used in various visual recognition systems, one of the first being that of Swain and Ballard (1991), where the intersection of global color histograms was used to compare images. Pyramids have been shown to be a useful representation in a wide variety of image processing tasks, from image coding (Burt and Adelson, 1983), to optical flow (Anandan, 1987), to texture modeling (Malik and Perona, 1990). See work by Hadjidemetriou et al. (2004) for a summary.

In the method of Indyk and Thaper (2003), multi-resolution histograms are compared with L_1 distance to approximate a least-cost matching of equal-mass global color histograms for nearest neighbor image retrievals. This work inspired our use of a similar representation for point sets and to consider counting matches within histograms. However, in contrast Indyk and Thaper's approach, our method builds a discriminative classifier or regressor over sets of local features, and it allows inputs to have unequal cardinalities. Most importantly, it enables partial matchings, which is important in practice for handling clutter and unsegmented images. In addition, we show that our approximate matching forms a valid Mercer kernel and explore its use for kernel-based learning for various applications.

In this work we develop a new kernel function that efficiently handles inputs that are unordered sets of varying sizes. We show how the pyramid match kernel may be used in conjunction with existing kernel-based learning algorithms to successfully learn decision boundaries or real-valued functions from the multi-set representation. Ours is the first work to show the histogram pyramid's connection to the optimal partial matching when used with a hierarchical weighted histogram intersection similarity measure. [1]

3. Approach

The main contribution of this work is a new kernel function based on implicit correspondences that enables discriminative classification and regression for unordered, variable-sized sets of vectors. The kernel is provably positive-definite. The main advantages of our algorithm are its efficiency, its use of the joint statistics of co-occurring features, and its resistance to clutter or "superfluous"

1. We first introduced the pyramid match kernel for the purpose of discriminative object recognition in a recent conference paper (Grauman and Darrell, 2005). This paper builds on that initial work: here we also demonstrate its utility for regression, give results on additional new data sets (one of which is non-vision), provide a more in-depth description of the kernel, and prove bounds on the error of the pyramid match relative to the optimal partial matching cost.

data points. The basic idea of our method is to map sets of features to multi-resolution histograms, and then compare the histograms with a weighted histogram intersection measure in order to approximate the similarity of the best partial matching between the feature sets. We call the proposed matching kernel the *pyramid match kernel* because input sets are converted to multi-resolution histograms.

3.1 Preliminaries

We consider a feature space F of d-dimensional vectors. The point sets (or multi-sets, since duplications of features may occur within a single set) we match will come from the input space S, which contains sets of feature vectors drawn from F:

$$S = \left\{ \mathbf{X} | \mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\} \right\},$$

where each feature is a d-dimensional vector, $\mathbf{x}_i \in F \subseteq \Re^d$, and $m = |\mathbf{X}|$. Note that the point dimension d is fixed for all features in F, but the value of m may vary across instances in S. The values of elements in vectors in F have a maximal range D, and the minimum inter-vector distance between unique points is 1, which may be enforced by scaling the data to some precision and truncating to integer values.

In this work we want to efficiently approximate the optimal partial matching. A partial matching between two point sets is an assignment that maps all points in the smaller set to some subset of the points in the larger (or equally-sized) set. Given point sets $\mathbf{X}$ and $\mathbf{Y}$, where $m = |\mathbf{X}|$, $n = |\mathbf{Y}|$, and $m \leq n$, a partial matching

$$\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi) = \{(\mathbf{x}_1, \mathbf{y}_{\pi_1}), \ldots, (\mathbf{x}_m, \mathbf{y}_{\pi_m})\}$$

pairs each point in $\mathbf{X}$ to some unique point in $\mathbf{Y}$ according to the permutation of indices specified by $\pi = [\pi_1, \ldots, \pi_m]$, $1 \leq \pi_i \leq n$, where π_i specifies which point $\mathbf{y}_{\pi_i} \in \mathbf{Y}$ is matched to $\mathbf{x}_i \in \mathbf{X}$, for $1 \leq i \leq m$. The cost of a partial matching is the sum of the distances between matched points:

$$\mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi)) = \sum_{\mathbf{x}_i \in \mathbf{X}} ||\mathbf{x}_i - \mathbf{y}_{\pi_i}||_1.$$

The optimal partial matching $\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)$ uses the assignment π^* that minimizes this cost:

$$\pi^* = \underset{\pi}{\operatorname{argmin}}\; \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi)). \qquad (1)$$

In order to form a kernel function based on correspondences, we are interested in evaluating partial matching *similarity*, where similarity is measured in terms of inverse distance or cost. The similarity $\mathcal{S}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi))$ of a partial matching is the sum of the inverse distances between matched points:

$$\mathcal{S}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi)) = \sum_{\mathbf{x}_i \in \mathbf{X}} \frac{1}{||\mathbf{x}_i - \mathbf{y}_{\pi_i}||_1 + 1},$$

where the distance in the denominator is incremented by 1 to avoid division by zero. We define the optimal partial matching similarity score to be the similarity resulting from the same matching that minimizes the cost (Eqn. 1).

3.2 The Pyramid Match Algorithm

The pyramid match approximation uses a multi-dimensional, multi-resolution histogram pyramid to partition the feature space into increasingly larger regions. At the finest resolution level in the pyramid, the partitions (bins) are very small; at successive levels they continue to grow in size until the point where a single partition encompasses the entire feature space. At some level along this gradation in bin sizes, any two points from any two point sets will begin to share a bin, and when they do, they are considered matched. The pyramid allows us to extract a matching score without computing distances between any of the points in the input sets—when points sharing a bin are counted as matched, the size of that bin indicates the farthest distance any two points in it could be from one another.

Each feature set is mapped to a multi-resolution histogram that preserves the individual features' distinctness at the finest level. The histogram pyramids are then compared using a weighted histogram intersection computation, which we show defines an implicit partial correspondence based on the finest resolution histogram cell where a matched pair first appears. The computation time of both the pyramids themselves as well as the weighted intersection is linear in the number of features.

The feature extraction function Ψ for an input set $\mathbf{X}$ is defined as:

$$\Psi(\mathbf{X}) = [H_0(\mathbf{X}), \ldots, H_{L-1}(\mathbf{X})], \tag{2}$$

where $\mathbf{X} \in S$, $L = \lceil \log_2 D \rceil + 1$, $H_i(\mathbf{X})$ is a histogram vector formed over points in $\mathbf{X}$ using d-dimensional bins of side length 2^i, and $H_i(\mathbf{X})$ has a dimension $r_i = \left(\frac{D}{2^i}\right)^d$. In other words, $\Psi(\mathbf{X})$ is a histogram pyramid, where each subsequent component histogram has bins that double in size (in all d dimensions) compared to the previous one. The bins in the finest-level histogram H_0 are small enough that each unique d-dimensional data point from features in F falls into its own bin, and then the bin size increases until all points in F fall into a single bin at level $L-1$.[2]

The pyramid match $\mathcal{P}_\Delta$ measures similarity (or dissimilarity) between point sets based on implicit correspondences found within this multi-resolution histogram space. The similarity between two input sets $\mathbf{Y}$ and $\mathbf{Z}$ is defined as the weighted sum of the number of feature matchings found at each level of the pyramid formed by Ψ:

$$\mathcal{P}_\Delta\left(\Psi(\mathbf{Y}), \Psi(\mathbf{Z})\right) = \sum_{i=0}^{L-1} w_i N_i, \tag{3}$$

where N_i signifies the number of newly matched pairs at level i, and w_i is a weight for matches formed at level i (and will be defined below). A new match is defined as a pair of features that were not in correspondence at any finer resolution level.

The matching approximation implicitly finds correspondences between point sets, if we consider two points matched once they fall into the same histogram bin, starting at the finest resolution level where each unique point is guaranteed to be in its own bin. The correspondences are *implicit* in that matches are counted and weighted according to their strength, but the specific pairwise links between points need not be individually enumerated. The matching is a hierarchical process: vectors not found to correspond at a fine resolution have the opportunity to be matched at coarser resolutions. For example, in Figure 3, there are two points matched at the finest scale, two new matches at

2. To enable accurate pyramid matching even with high-dimensional feature spaces, we have developed a variant where the pyramid bin structure depends on the distribution of the data (Grauman and Darrell, 2007).

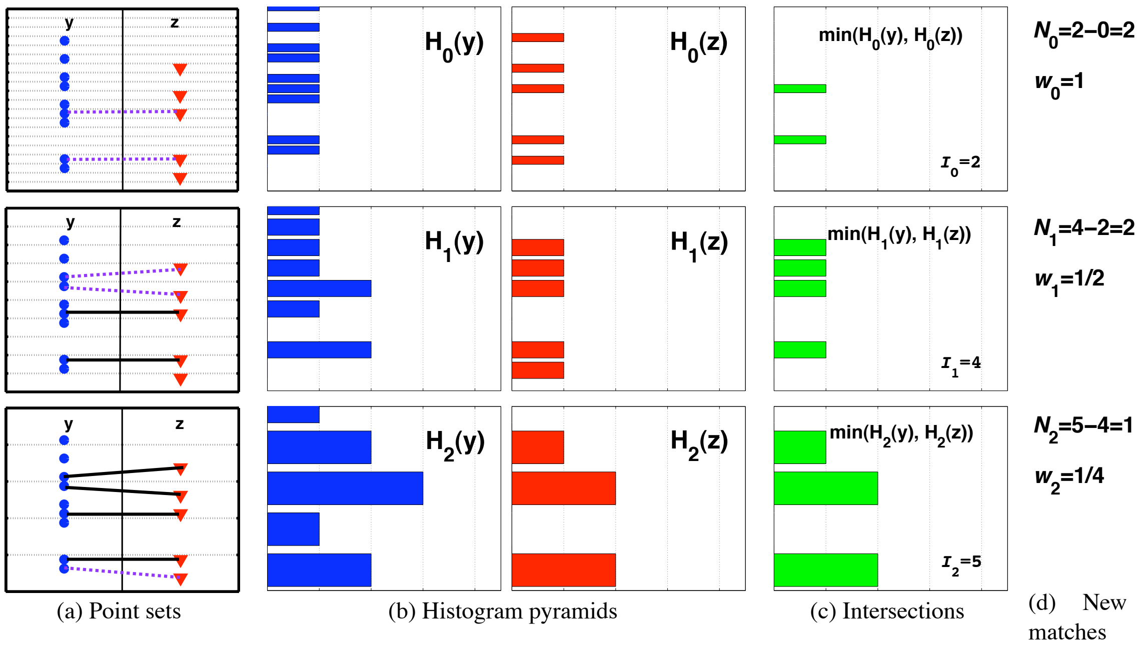


Figure 3: The pyramid match ($\mathcal{P}_\Delta$) determines a partial correspondence by matching points once they fall into the same histogram bin. In this example, two 1-D feature sets are used to form two histogram pyramids. Each row corresponds to a pyramid level. In (a), the set $\mathbf{Y}$ is on the left side, and the set $\mathbf{Z}$ is on the right. (Points are distributed along the vertical axis, and these same points are repeated at each level.) Light dotted lines are bin boundaries, bold dashed lines indicate a new pair matched at this level, and bold solid lines indicate a match already formed at a finer resolution level. In (b) multi-resolution histograms are shown, with bin counts along the horizontal axis. In (c) the intersection pyramids between the histograms in (b) are shown. $\mathcal{P}_\Delta$ uses these intersection counts to measure how many new matches occurred at each level. Here, $I_i = I(H_i(\mathbf{Y}), H_i(\mathbf{Z})) = 2, 4, 5$ across levels, and therefore the number of new matches found at each level are $N_i = 2, 2, 1$. ($I_{-1} = 0$ by definition.) The sum over N_i, weighted by $w_i = 1, \frac{1}{2}, \frac{1}{4}$, gives the pyramid match similarity.

the medium scale, and one at the coarsest scale. $\mathcal{P}_\Delta$'s output value reflects the overall similarity of the matching: each newly matched pair at level i contributes a value w_i that is proportional to how similar two points matching at that level must be, as determined by the bin size.

To calculate N_i, the pyramid match makes use of a histogram intersection function I, which measures the "overlap" between two histograms' bin counts:

$$I(\mathbf{A}, \mathbf{B}) = \sum_{j=1}^{r} \min\left(\mathbf{A}^{(j)}, \mathbf{B}^{(j)}\right),$$

where $\mathbf{A}$ and $\mathbf{B}$ are histograms with r bins, and $\mathbf{A}^{(j)}$ denotes the count of the j^{th} bin of $\mathbf{A}$.

Histogram intersection effectively counts the number of points in two sets that match at a given quantization level, that is, fall into the same bin. To calculate the number of newly matched pairs N_i induced at level i, it is sufficient to compute the difference between successive histogram levels' intersections:

$$N_i = I\left(H_i(\mathbf{Y}), H_i(\mathbf{Z})\right) - I\left(H_{i-1}(\mathbf{Y}), H_{i-1}(\mathbf{Z})\right), \tag{4}$$

where H_i refers to the i^{th} component histogram generated by Ψ in Eqn. 2. Note that the measure is not searching explicitly for similar points—it never computes distances between the vectors in each set. Instead, it simply uses the change in intersection values at each histogram level to count the matches as they occur. In addition, due to the subtraction in Eqn. 4, the output score will reflect an underlying matching that is one-to-one.

The sum in Eqn. 3 starts with index $i = 0$ because by the definition of F, the same points that could match at level 0 can match at (a hypothetical) level -1. Therefore we can start collecting new match counts at level 0, and define level -1 to be a base case with no intersections: $I\left(H_{-1}(\mathbf{Y}), H_{-1}(\mathbf{Z})\right) = 0$. All matches formed at level 0 are new.

The number of new matches found at each level in the pyramid is weighted according to the size of that histogram's bins: to measure similarity, matches made within larger bins are weighted less than those found in smaller bins. Specifically, we make a geometric bound of the distance between any two points sharing a particular bin; in terms of L_1 cost, two points in the same bin can only be as far from one another as the sum of the lengths of the bin's sides. At level i in a pyramid, this length is equal to $d2^i$. Thus, the number of new matches induced at level i is weighted by $w_i = \frac{1}{d2^i}$ to reflect the (worst-case) similarity of points matched at that level. Intuitively, this means that similarity between vectors (features in $\mathbf{Y}$ and $\mathbf{Z}$) at a finer resolution—where features are more distinct—is rewarded more heavily than similarity between vectors at a coarser level.[3] Moreover, as we show in Section 6, this particular setting of the weights enables us to prove theoretical error bounds for the pyramid match cost.

From Eqns. 3 and 4, we define the (un-normalized) pyramid match:

$$\tilde{\mathcal{P}}_\Delta\left(\Psi(\mathbf{Y}), \Psi(\mathbf{Z})\right) = \sum_{i=0}^{L-1} w_i \Big(I\left(H_i(\mathbf{Y}), H_i(\mathbf{Z})\right) - I(H_{i-1}(\mathbf{Y}), H_{i-1}(\mathbf{Z}))\Big), \tag{5}$$

where $\mathbf{Y}, \mathbf{Z} \in S$, and $H_i(\mathbf{Y})$ and $H_i(\mathbf{Z})$ refer to the i^{th} histogram in $\Psi(\mathbf{Y})$ and $\Psi(\mathbf{Z})$, respectively. We normalize this value by the product of each input's self-similarity to avoid favoring larger input sets, arriving at the final kernel value $\mathcal{P}_\Delta(\mathbf{P}, \mathbf{Q}) = \frac{1}{\sqrt{C}} \tilde{\mathcal{P}}_\Delta(\mathbf{P}, \mathbf{Q})$, where $C = \tilde{\mathcal{P}}_\Delta(\mathbf{P}, \mathbf{P})\, \tilde{\mathcal{P}}_\Delta(\mathbf{Q}, \mathbf{Q})$.

In order to alleviate quantization effects that may arise due to the discrete histogram bins, we combine the kernel values resulting from multiple (T) pyramid matches formed under different multi-resolution histograms with randomly shifted bins. Each dimension of each of the T pyramids is shifted by an amount chosen uniformly at random from $[0, D]$. This yields T feature mappings $\Psi_1, \ldots, \Psi_T$ that are applied as in Eqn. 2 to map an input set $\mathbf{X}$ to T multi-resolution histograms: $[\Psi_1(\mathbf{X}), \ldots, \Psi_T(\mathbf{X})]$. For inputs $\mathbf{Y}$ and $\mathbf{Z}$, the combined kernel value is then $\sum_{j=1}^{T} \mathcal{P}_\Delta\left(\Psi_j(\mathbf{Y}), \Psi_j(\mathbf{Z})\right)$.

To further refine a pyramid match, multiple pyramids with unique initial (finest level) bin sizes may be used. The set of unique bin sizes produced throughout a pyramid determines the range

3. This is if the pyramid match is a kernel measuring similarity; an inverse weighting scheme is applied if the pyramid match is used to measure cost. To use the matching as a cost function, weights are set as the distance estimates ($w_i = d2^i$); to use as a similarity measure, weights are set as (some function of) the inverse of the distance estimates ($w_i \propto \frac{1}{d2^i}$).

of distances at which features will be considered for possible matchings. With bin sizes doubling in size at every level of a pyramid, this means that the range of distances considered will rapidly increase until the full diameter of the feature space is covered. However, by incorporating multiple pyramids with unique initial bin sizes, we can boost that range of distances to include more diverse gradations. This is accomplished by setting the side length for each histogram $H_i(\mathbf{X})$ to $f2^i$, where f is the side length at the finest resolution (in the above definition, $f = 1$). For example, say $D = 32$, so $L = 6$. Then the set of distances considered with $f = 1$ are $\{1,2,4,8,16,32\}$; adding a second pyramid with $f = 3$ increases this set to also consider the distances $\{3,6,12,24\}$. The outputs from pyramids with multiple starting side lengths are combined in the same way that the outputs from pyramids with multiple translations are combined.

In fact, the rationale for considering multiple random shifts of the pyramid grids is not only to satisfy the intuitive need to reduce quantization effects caused by bin placements; it also allows us to theoretically measure how probable it is (in the expectation) that any two points will be separated by a bin boundary, as we describe below in Section 6.

3.3 Partial Match Correspondences

The pyramid match allows input sets to have unequal cardinalities, and therefore it enables partial matchings, where the points of the smaller set are mapped to some subset of the points in the larger set. Dissimilarity is only judged on the most similar part of the empirical distributions, and superfluous data points from the larger set are entirely ignored; the result is a robust similarity measure that accommodates inputs expected to contain extraneous vector entries. This is a common situation when recognizing objects in images, due for instance to background variations, clutter, or changes in object pose that cause different subsets of features to be visible. As a kernel, the proposed matching measure is equipped to handle unsegmented or poorly segmented examples, as we will demonstrate in Section 9.

Since the pyramid match defines correspondences across entire sets simultaneously, it inherently accounts for the distribution of features occurring in one set. In contrast, previous approaches have used each feature in a set to independently index into the second set; this ignores possibly useful information that is inherent in the co-occurrence of a set of distinctive features, and it fails to distinguish between instances where an object has varying numbers of similar features since multiple features may be matched to a single feature in the other set (Wallraven et al., 2003; Lyu, 2005; Boughorbel et al., 2005; Lowe, 2004; Mikolajczyk and Schmid, 2001; Tuytelaars and Gool, 1999; Shaffalitzky and Zisserman, 2002).

4. Efficiency

A key aspect of this method is that we obtain a measure of matching quality between two point sets without computing pairwise distances between their features—an $O(m^2)$ savings over sub-optimal greedy matchings. Instead, we exploit the fact that the points' placement in the pyramid reflects their distance from one another in the feature space.

The time required to compute the L-level histogram pyramid $\Psi(\mathbf{X})$ for an input set with $m = |\mathbf{X}|$ d-dimensional features is $O(dzL)$, where $z = \max(m,k)$ and k is the maximum histogram index value in a single dimension. For a histogram at level i, $k \leq \frac{D}{2^i}$, so at any level, $k \leq D$. (Typically $m > k$.) The bin coordinates corresponding to nonzero histogram entries for each of the $L = \lceil \log_2 D \rceil + 1$ quantization levels are computed directly during a scan of the m input vectors; these entries are sorted by the bin indices and the bin counts for all entries with the same index are summed to form one entry. This sorting requires only $O(dm+dk)$ time using the radix-sort algorithm with counting sort, a linear time sorting algorithm that is applicable to the integer bin indices Cormen et al. (1990). The histogram pyramid that results is high-dimensional, but very sparse, with only $O(mL)$ nonzero entries that need to be stored.

The computational complexity of $\mathcal{P}_\Delta$ is $O(dmL)$, since computing the intersection values for histograms that have been sorted by bin index requires time linear in the number of nonzero entries (*not* the number of actual bins). With sorted bin index lists, we obtain the intersection value by running one pointer down each of the two lists; one pointer may not advance until the other is incremented down to an index at least as high as the other, or else the end of its list. Then, whenever the two lists share a nonzero index, the intersection count is incremented by the minimum bin count between the two. If one list has a nonzero count for some bin index but the other has no entry for that index, the minimum count is 0, so no update is needed. In the applications we explore in our experiments, typical values of the variables affecting complexity are as follows: $5 \leq d \leq 12$, $300 \leq m \leq 3000$, and $D \approx 250$, which yields $L = 9$.

Generating multiple pyramid matches with randomly shifted grids scales the complexity by T, the constant number of shifts. (Typically we will use $1 \leq T \leq 3$.) All together, the complexity of computing both the pyramids and kernel or cost values is $O(TdmL)$. In contrast, the optimal matching requires $O(dm^3)$ time, which severely limits the practicality of large input sizes. Refer to Figure 2 for complexity comparisons with existing set kernels.

5. Satisfying Mercer's Condition

Kernel-based learning algorithms are founded on the idea of embedding data into a Euclidean space, and then seeking linear relations among the embedded data (Shawe-Taylor and Cristianini, 2004; Vapnik, 1998). For example, a support vector machine (SVM) finds the optimal separating hyperplane between two classes in an embedded space (also referred to as the feature space). A kernel function $K : X \times X \rightarrow \Re$ serves to map pairs of data objects in an input space X to their inner product in the embedding space E, thereby evaluating the similarities between all data objects and determining their relative positions. Linear relations are sought in the embedded space, but the learned function may still be non-linear in the input space, depending on the choice of a feature mapping function $\Phi : X \rightarrow E$.

Only positive semi-definite kernels guarantee an optimal solution to kernel-based algorithms based on convex optimization, which includes SVMs. According to Mercer's theorem, a kernel K is positive semi-definite if and only if there exists a mapping Φ such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle, \ \forall \mathbf{x}_i, \mathbf{x}_j \in X,$$

where $\langle \cdot, \cdot \rangle$ denotes a scalar dot product. This insures that the kernel corresponds to an inner product in some feature space, where kernel methods can search for linear relationships (Shawe-Taylor and Cristianini, 2004).

Proposition 1

The pyramid match yields a Mercer kernel.

Proof:

Histogram intersection on single resolution histograms over multi-dimensional data was shown to be a positive-definite function by Odone et al. (2005). That is, the intersection $I(H(\mathbf{Y}),H(\mathbf{Z}))$ is a Mercer kernel. The proof shows that there is an explicit feature mapping after which the intersection is an inner product. Specifically, the mapping $\mathcal{V}$ encodes an r-bin histogram H as a p-dimensional binary vector, $p = m \times r$, where m is the total number of points in the histogram:

$$\mathcal{V}(H) = \left(\underbrace{\overbrace{1,\ldots,1}^{H^{(1)}},\ \overbrace{0,\ldots,0}^{m-H^{(1)}}}_{\text{first bin}},\ldots,\underbrace{\overbrace{1,\ldots,1}^{H^{(r)}},\ \overbrace{0,\ldots,0}^{m-H^{(r)}}}_{\text{last bin}} \right).$$

The inner product between the binary strings output from $\mathcal{V}$ is equivalent to the original histograms' intersection value (Odone et al., 2005). If m varies across examples, the above holds by setting $p = M \times r$, where M is the maximum size of any input. Note that this binary encoding only serves to prove positive-definiteness and is never computed explicitly.

Using this proof and the closure properties of valid kernel functions, we can show that the pyramid match is a Mercer kernel. The definition given in Eqn. 5 is algebraically equivalent to

$$\tilde{\mathcal{P}}_\Delta(\Psi(\mathbf{Y}),\Psi(\mathbf{Z})) = w_{L-1}\ I\left(H_{L-1}(\mathbf{Y}),H_{L-1}(\mathbf{Z})\right) + \sum_{i=0}^{L-2}(w_i - w_{i+1})\ I\left(H_i(\mathbf{Y}),H_i(\mathbf{Z})\right),$$

since $I\left(H_{-1}(\mathbf{Y}),H_{-1}(\mathbf{Z})\right) = 0$ by definition. Given that Mercer kernels are closed under both addition and scaling by a positive constant (Shawe-Taylor and Cristianini, 2004), the above form shows that the pyramid match kernel is positive semi-definite for any weighting scheme where $w_i \geq w_{i+1}$. In other words, if we can insure that the weights decrease for coarser pyramid levels, then the pyramid match will sum over positively weighted positive-definite kernels, yielding another positive-definite kernel. Using the weights $w_i = \frac{1}{d2^i}$, we do maintain this property. The sum combining the outputs from multiple pyramid matches under different random bin translations likewise remains Mercer. Therefore, the pyramid match is valid for use as a kernel in any existing learning methods that require Mercer kernels.

6. Approximation Error Bounds

In this section we show theoretical approximation bounds on the cost measured by the pyramid match relative to the cost measured by the optimal partial matching defined in Section 3.1. Recall the cost (as opposed to similarity) is measured by setting $w_i = d2^i$ in Eqn. 5. In earlier work, Indyk and Thaper (2003) provided bounds for a multi-resolution histogram embedding's approximation of the optimal bijective matching between inputs with equal cardinality. Some of the main ideas of the proofs below are similar, but they have been adapted and extended to show the expected error bounds for partial matchings, where input sets can have variable numbers of features, and some features do not affect the matching cost.

Proposition 2

For any two point sets $\mathbf{X}$,$\mathbf{Y}$ where $|\mathbf{X}| \leq |\mathbf{Y}|$, we have

$$\mathcal{C}\left(\mathcal{M}(\mathbf{X},\mathbf{Y};\pi^*)\right) \leq \mathcal{P}_\Delta(\mathbf{X},\mathbf{Y}).$$

Proof:

Consider a matching induced by pairing points within the same cells of each histogram H_i. The cost induced by matching two points within a bin having d sides that are each of length 2^i is at most $d2^i$ under the L_1 ground distance.

There are no pairings induced by the histograms at level -1, so $\mathcal{I}(H_{-1}(\mathbf{X}), H_{-1}(\mathbf{Y})) = 0$. At level 0 there are $\mathcal{I}(H_0(\mathbf{X}), H_0(\mathbf{Y}))$ pairs of points that can be matched together within the same bins of H_0, which induces a cost no more than $d\ \mathcal{I}(H_0(\mathbf{X}), H_0(\mathbf{Y}))$. Then $\mathcal{I}(H_1(\mathbf{X}), H_1(\mathbf{Y})) - \mathcal{I}(H_0(\mathbf{X}), H_0(\mathbf{Y}))$ new pairs of points are matched at level 1, which induces a cost no more than $2d\left[\mathcal{I}(H_1(\mathbf{X}), H_1(\mathbf{Y})) - \mathcal{I}(H_0(\mathbf{X}), H_0(\mathbf{Y}))\right]$, and so on.

In general, histogram H_i induces cost $d2^i\left[\mathcal{I}(H_i(\mathbf{X}), H_i(\mathbf{Y})) - \mathcal{I}(H_{i-1}(\mathbf{X}), H_{i-1}(\mathbf{Y}))\right]$. Summing the costs induced by all levels, we have:

$$\begin{aligned}
\mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)) &\leq \sum_{i=0}^{L-1} d2^i \Big(\mathcal{I}(H_i(\mathbf{X}), H_i(\mathbf{Y})) - \mathcal{I}(H_{i-1}(\mathbf{X}), H_{i-1}(\mathbf{Y})) \Big) \\
&\leq \mathcal{P}_\Delta(\mathbf{X}, \mathbf{Y}),
\end{aligned}$$

where $w_i = d2^i$ in Eqn. 5.

Proposition 3

There is a constant C such that for any two point sets $\mathbf{X}$ and $\mathbf{Y}$, where $|\mathbf{X}| \leq |\mathbf{Y}|$, if the histogram bin boundaries are shifted randomly in the same way when computing $\Psi(\mathbf{X})$ and $\Psi(\mathbf{Y})$, then the expected value of the pyramid match cost is bounded:

$$E[\mathcal{P}_\Delta(\mathbf{X}, \mathbf{Y})] \leq \Big(C \cdot d \log D + d \Big)\ \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)).$$

Proof:

To write the pyramid match in terms of counts of *unmatched* points in the implicit partial matching, we define the *directed distance* between two histograms formed from point sets $\mathbf{X}$ and $\mathbf{Y}$:

$$\begin{aligned}
\mathcal{D}(H(\mathbf{X}), H(\mathbf{Y})) &= \sum_{j=1}^{r} \mathcal{D}_j\left(H(\mathbf{X})^{(j)}, H(\mathbf{Y})^{(j)} \right) \text{, where} \\
\mathcal{D}_j(a, b) &= \begin{cases} a - b, & \text{if } a > b \\ 0, & \text{otherwise.} \end{cases}
\end{aligned}$$

The directed distance $\mathcal{D}(H_i(\mathbf{X}), H_i(\mathbf{Y}))$ counts the number of unmatched points at resolution level i. Note that $\mathcal{D}(H_{-1}(\mathbf{X}), H_{-1}(\mathbf{Y})) = |\mathbf{X}|$ and $\mathcal{D}(H_{L-1}(\mathbf{X}), H_{L-1}(\mathbf{Y})) = 0$ by definition. The directed distance value decreases with i, as bins are increasing in size and more implicit matches are made. The change in the directed distance across levels is the decrease in the count of unmatched points from one level to the next, and therefore also serves to count the number of new matches formed at level i:

$$\mathcal{I}(H_i(\mathbf{X}), H_i(\mathbf{Y})) - \mathcal{I}(H_{i-1}(\mathbf{X}), H_{i-1}(\mathbf{Y})) = \mathcal{D}(H_{i-1}(\mathbf{X}), H_{i-1}(\mathbf{Y})) - \mathcal{D}(H_i(\mathbf{X}), H_i(\mathbf{Y})).$$

In terms of the directed distance, the pyramid match cost is

$$\begin{aligned}\mathcal{P}_\Delta(\mathbf{X},\mathbf{Y}) &= \sum_{i=0}^{L-1} d2^i \Big(\mathcal{D}(H_{i-1}(\mathbf{X}),H_{i-1}(\mathbf{Y})) - \mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y}))\Big) \\ &= d\mathcal{D}(H_{-1}(\mathbf{X}),H_{-1}(\mathbf{Y})) + \sum_{i=0}^{L-2} d2^i\ \mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y})) \\ &= d|\mathbf{X}| + \sum_{i=0}^{L-2} d2^i\ \mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y})).\end{aligned}$$

The expected value of the pyramid match cost is then

$$E\left[\mathcal{P}_\Delta(\mathbf{X},\mathbf{Y})\right] = d|\mathbf{X}| + \sum_{i=0}^{L-2} d2^i\ E\left[\mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y}))\right]. \tag{6}$$

The optimal partial matching $\mathcal{M}(\mathbf{X},\mathbf{Y};\pi^*) = \left\{(\mathbf{x}_1,\mathbf{y}_{\pi_1^*}),\ldots,(\mathbf{x}_m,\mathbf{y}_{\pi_m^*})\right\}$ implies a graph whose nodes are those points in the input sets $\mathbf{X}$ and $\mathbf{Y}$ that participate in the matching (i.e., all points in $\mathbf{X}$ and a subset from $\mathbf{Y}$), and whose edges connect the matched pairs $(\mathbf{x}_i,\mathbf{y}_{\pi_i})$, for $1 \le i \le |\mathbf{X}|$. Let n_j be the number of edges in this graph that have lengths in the range $[d2^{j-1}, d2^j)$. A bound on the optimal partial matching may then be expressed as:

$$\begin{aligned}\sum_j n_j\ d2^{j-1} &\le \mathcal{C}(\mathcal{M}(\mathbf{X},\mathbf{Y};\pi^*)), \text{or} \\ \sum_j n_j\ d2^j &\le 2\ \mathcal{C}(\mathcal{M}(\mathbf{X},\mathbf{Y};\pi^*)),\end{aligned} \tag{7}$$

since for every j, all n_j edges must be of length at least $d2^{j-1}$.

The directed distance $\mathcal{D}$ counts at a given resolution how many points from set $\mathbf{X}$ are unmatched. Any edge in the optimal matching graph that is left "uncut" by the bin boundaries in histogram H_i contributes nothing to $\mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y}))$. Any edge that is "cut" by the histogram contributes at most 1 to $\mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y}))$. Therefore, the expected value of the directed distance at a given resolution is bounded by:

$$E\left[\mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y}))\right] \le \sum_j E\left[T_{ij}\right], \tag{8}$$

where T_{ij} is the number of edges in the optimal matching having lengths in the range $[d2^{j-1}, d2^j)$ that are cut by H_i.

The probability $\Pr(\text{cut}(\mathbf{x},\mathbf{y});i)$ that an edge for $(\mathbf{x},\mathbf{y})$ in the optimal matching is cut by the histogram grid H_i is bounded by $\frac{||\mathbf{x}-\mathbf{y}||_1}{2^i}$. This can be seen by considering the probability of a bin boundary cutting an edge in any one of the d dimensions, taking into account that the bin boundaries are shifted independently in each dimension. An edge with length $||\mathbf{x}-\mathbf{y}||_1 > d2^i$ is certainly cut by the histogram grid at H_i. An edge with length $||\mathbf{x}-\mathbf{y}||_1 \le d2^i$ may be cut in any dimension. In this case, the probability of a cut in dimension k for points $\mathbf{x} = [x_1,\ldots,x_d], \mathbf{y} = [y_1,\ldots,y_d]$ is $\frac{|x_k-y_k|}{2^i}$, for $1 \le k \le d$. The probability of a cut from H_i occurring in some dimension is then bounded by the sum of these probabilities:

$$\begin{aligned}\Pr(\text{cut}(\mathbf{x},\mathbf{y});i) &\le \sum_{k=1}^{d} \frac{|x_k-y_k|}{2^i} \\ &\le \frac{||\mathbf{x}-\mathbf{y}||_1}{2^i}.\end{aligned}$$

This provides an upper bound on the expected number of edges in the optimal matching that are cut at level i:

$$E\left[T_{ij}\right] \leq n_j \, \frac{d2^j}{2^i},$$

since by definition $d2^j$ is the maximum distance between any pair of matched points counted by n_j.

Now with Eqns. 7 and 8 we have a bound for the expected directed distance at a certain resolution level:

$$E\left[\mathcal{D}\left(H_i(\mathbf{X}), H_i(\mathbf{Y})\right)\right] \leq \frac{1}{2^i} \, \sum_j n_j \, d2^j \leq \frac{1}{2^i} \, 2 \; \mathcal{C}\left(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)\right).$$

Relating this to the expected value of the pyramid match cost in Eqn. 6, we have

$$\begin{aligned} E\left[\mathcal{P}_\Delta(\mathbf{X}, \mathbf{Y})\right] &\leq d|\mathbf{X}| + \sum_{i=0}^{L-2} d2^i \left(\frac{1}{2^i} \, 2 \; \mathcal{C}\left(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)\right)\right) \\ &\leq d|\mathbf{X}| + 2d \, (L-1) \; \mathcal{C}\left(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)\right) \\ &\leq d|\mathbf{X}| + 2d \log D \; \mathcal{C}\left(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)\right) \qquad (9) \end{aligned}$$

since $L = \lceil \log_2 D \rceil + 1$.

As described in Section 3.1, points in sets $\mathbf{X}$ and $\mathbf{Y}$ are comprised of integer entries, insuring that the minimum inter-feature distance between unique points is 1. We also know that for the sake of measuring matching cost, $\mathbf{X}$ and $\mathbf{Y}$ are disjoint sets: $\mathbf{X} \cap \mathbf{Y} = \emptyset$; if there are any identical points in the two sets, we can consider them as discarded (in pairs) to produce strictly disjoint sets, since identical points contribute nothing to the matching cost. Since $\mathbf{X}, \mathbf{Y} \subseteq [D]^d$ and $\mathbf{X} \cap \mathbf{Y} = \emptyset$, we have $|\mathbf{X}| \leq \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*))$. That is, each edge in the optimal matching must have a length of at least 1, making the number of points in the smaller set a lower bound on the cost of the optimal matching for any two sets. With this bound and Eqn. 9, we have the following bound on the expected pyramid match cost error

$$\begin{aligned} E\left[\mathcal{P}_\Delta(\mathbf{X}, \mathbf{Y})\right] &\leq d \; \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)) + 2d \log D \; \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)) \\ &\leq \left(2d \log D + d\right) \; \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)) \\ &\leq \left(C \cdot d \log D + d\right) \; \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)). \end{aligned}$$

7. Classification and Regression with the Pyramid Match

We train support vector machines and support vector regressors (SVRs) to perform classification and regression with the pyramid match kernel. An SVM or SVR is trained by specifying the matrix of kernel values between all pairs of training examples. The kernel's similarity values determine the examples' relative positions in an embedded space, and quadratic programming is used to find the optimal separating hyperplane or function between the two classes in this space. Because the pyramid match kernel is positive-definite we are guaranteed to find a unique optimal solution.

We have found that the pyramid match kernel can produce kernel matrices with dominant diagonals, particularly as the dimension of the features in the sets increases. The reason for this is that as the dimension of the points increases, there are a greater number of finer-resolution histogram levels in the pyramid where two input sets will have few shared bins. Once the quantization level is coarse enough, two sets will start having significant histogram intersection values. However, these intersections will receive lower weights due to the $\frac{1}{d2^i}$ weighting scheme, which by construction accurately reflects the maximal distance between points falling into the same bin at level i. On the other hand, an input set compared against itself will result in large histogram intersection values at each level—specifically intersection values equal to the number of points in the set, which after normalization generates a diagonal entry of one.

The danger of having a kernel matrix diagonal that is significantly larger than the off-diagonal entries is that the examples appear nearly orthogonal in the feature space, in some cases causing an SVM or SVR to essentially "memorize" the data and impairing its sparsity and generalization ability (Shawe-Taylor and Cristianini, 2004). Nonetheless, we are able to work around this issue by modifying the initial kernel matrix in such a way that reduces its dynamic range, while preserving its positive-definiteness. We use the functional calculus transformation suggested in Weston et al. (2002): a subpolynomial kernel is applied to the original kernel values, followed by an empirical kernel mapping that embeds the distance measure into a feature space. Thus, when necessary to reduce diagonal dominance, first kernel values $\mathbf{K}_{ij}$ generated by $\mathcal{P}_\Delta$ are updated to $\mathbf{K}_{ij} \leftarrow \mathbf{K}_{ij}{}^p$, $0 < p < 1$. Then the kernel matrix $\mathbf{K}$ is replaced with $\mathbf{K}\mathbf{K}^\mathrm{T}$ to obtain the empirical feature map $\Phi_e(\mathbf{y}) = [K(\mathbf{y}, \mathbf{x}_1), \ldots, K(\mathbf{y}, \mathbf{x}_N)]^\mathrm{T}$ for N training examples. As in Weston et al. (2002), the parameter p is chosen with cross-validation. This post-processing of the kernel matrix is not always necessary; both the dimension of the points as well as the specific structure of a given data set will determine how large the initial kernel matrix diagonal is.

8. Empirical Quality of Approximate Partial Matchings

The approximation bounds we can show are actually significantly weaker than what we observe in practice. In this section, we empirically evaluate the approximation quality of the pyramid match, and make a direct comparison between our partial matching approximation and the L_1 embedding of Indyk and Thaper (2003).

We conducted experiments to evaluate how close the correspondences implicitly measured by the pyramid match are to the true optimal correspondences—the matching that results in the minimal summed cost between corresponding points. In order to work with realistic data but still have control over the sizes of the sets and the amount of clutter features, we established synthetic "category" models. Each model is comprised of some fixed number m' of parts, and each part has a Gaussian model that generates its d-dimensional appearance vector (in the spirit of the "constellation model" used by Fergus et al., 2003, and others). Given these category models, we can then add clutter features, adjust noise parameters, and so on, simulating in a controlled manner the variations that occur with the sets of image patches extracted from an actual object. The appearance of the clutter features is determined by selecting a random vector from a uniform distribution on the same range of values as the model features.

We generated two data sets, one with equally-sized sets, and one with variable-sized sets. Every point set was drawn from one of two synthetic category models, with $m' = 35$ and $d = 2$. For the first data set, 50 point sets containing only the m' model features were sampled from both of the two category models, for a total of 100 equally-sized point sets. For the other data set, the model feature sets were merged with a randomly generated number C of "extra" clutter features, for a total of 100 point sets with $m' + C$ features each, with C selected uniformly at random from $[0, 100]$. We compared the pyramid match's outputs to those produced by the optimal partial matching obtained via a linear programming solution to the transportation problem (Rubner et al., 2000), as well as those produced by an L_1 approximation (Indyk and Thaper, 2003). For both of the data sets, we computed the pairwise set-to-set distances using each of these three measures.

If an approximate measure is approximating the optimal matching well, we should find the ranking induced by that approximation to be highly correlated with the ranking produced by the optimal matching for the same data. In other words, the point sets should be sorted similarly by either method. We can display results in two ways to evaluate if this is true: 1) by plotting the actual costs computed by the optimal and approximate method, and 2) by plotting the rankings induced by the optimal and approximate method. Spearman's rank correlation coefficient R provides a good quantitative measure to evaluate the ranking consistency:

$$R = 1 - \frac{6\sum_{i=1}^{N}(i - \hat{r}(i))^2}{N(N^2-1)},$$

where i is the rank value in the true order and $\hat{r}(i)$ is the corresponding rank assigned in the approximate ordering, for each of the N corresponding ordinal values assigned by the two measures.

Figure 4 displays both types of plots for the two data sets: the top row (a) displays plots corresponding to the data set with equally-sized sets, that is, for the bijective matching problem, while the bottom row (b) displays plots corresponding to the data set with variable-sized sets, that is, for the partial matching problem.

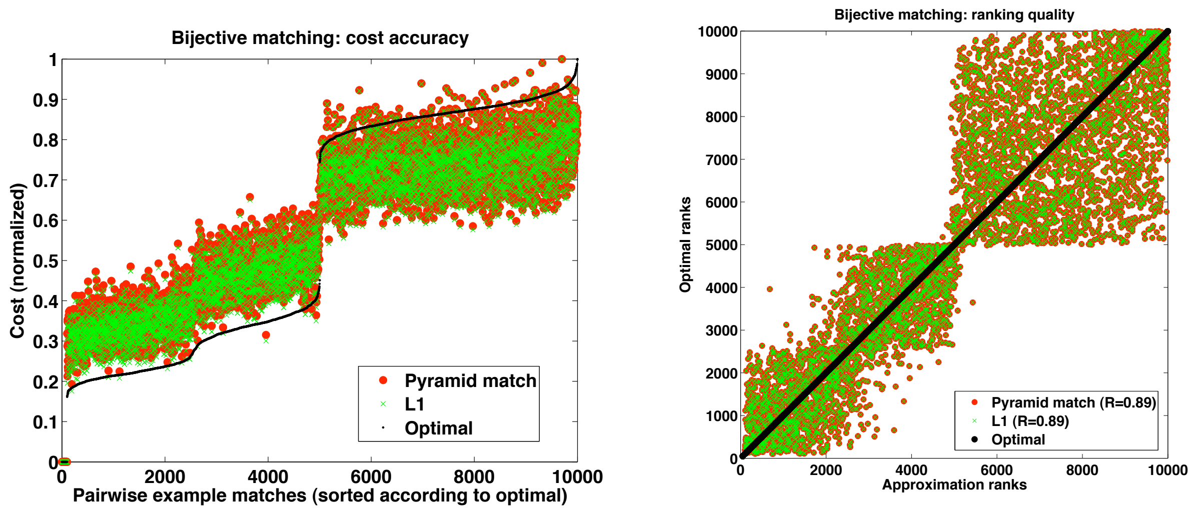
The two plots in the lefthand column show the normalized output costs from 10,000 pairwise set-to-set comparisons computed by the optimal matching (black), the pyramid match with the number of random shifts $T = 3$ (red circles), and the L_1 approximation, also with $T = 3$ (green x's). Note that in these figures we plot cost (so the pyramid match weights are set to $w_i = d2^i$), and for display purposes the costs have been normalized by the maximum cost produced for each measure to produce values between $[0,1]$. The cost values are sorted according to the optimal measure's magnitudes for visualization purposes. The raw values produced by the approximate measures will always overestimate the cost; normalizing by the maximum cost value simply allows us to view them against the optimal measure on the same scale.

The two plots in the righthand column display the rankings for each approximation plotted against the optimal rankings. The black diagonals in these plots denote the optimal performance, where the approximate rankings would be identical to the optimal ones. The R values displayed in the legends refer to the Spearman rank correlation scores for each approximation in that experiment; higher Spearman correlations have points clustered more tightly along this diagonal.

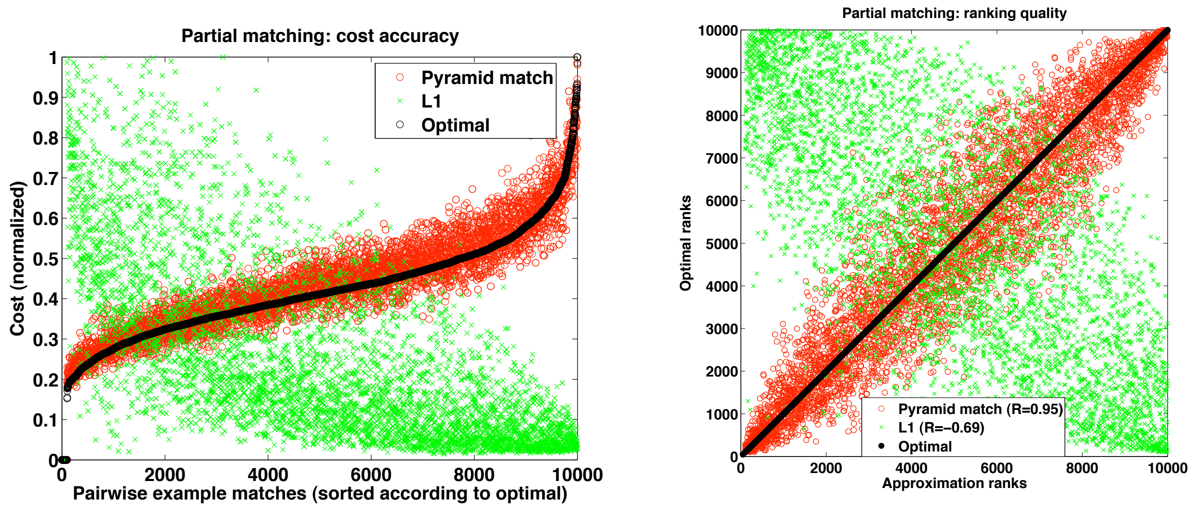
Both approximations compute equivalent costs and rankings for the bijective matching case, as indicated by Figure 4 (a). This is an expected outcome, since the L_1 distance over histograms is directly related to histogram intersection *if* those histograms have equal masses (Swain and Ballard, 1991), as they do for the bijective matching test:

$$I(H(\mathbf{Y}),H(\mathbf{Z})) = m - \frac{1}{2}||H(\mathbf{Y}) - H(\mathbf{Z})||_{L_1}, \text{ if } m = |\mathbf{Y}| = |\mathbf{Z}|.$$

The structure along the diagonal in the top right plot reflects the fact that two types (categories) of point sets were present in the data, causing items of the same category to block together when they are sorted by matching cost. The square-shaped cluster in the upper right portions of the plots show that while the approximate measures do not distinguish between all examples within a category precisely the way the optimal measure does, they do both consider all items within one category to be most distant from an example drawn from the other category. Similarly, there are discontinuities



(a) Bijective matching



(b) Partial matching

Figure 4: Pyramid match and L_1 embedding comparison on (a) bijective matchings with equally-sized sets and (b) partial matchings with variably-sized sets. When all input sets are the same size, the pyramid match and the L_1 embedding give equivalent results. However, the pyramid match also approximates the optimal correspond ences for input sets of unequal cardinalities and allows partial matches. (This figure is best viewed in color.)

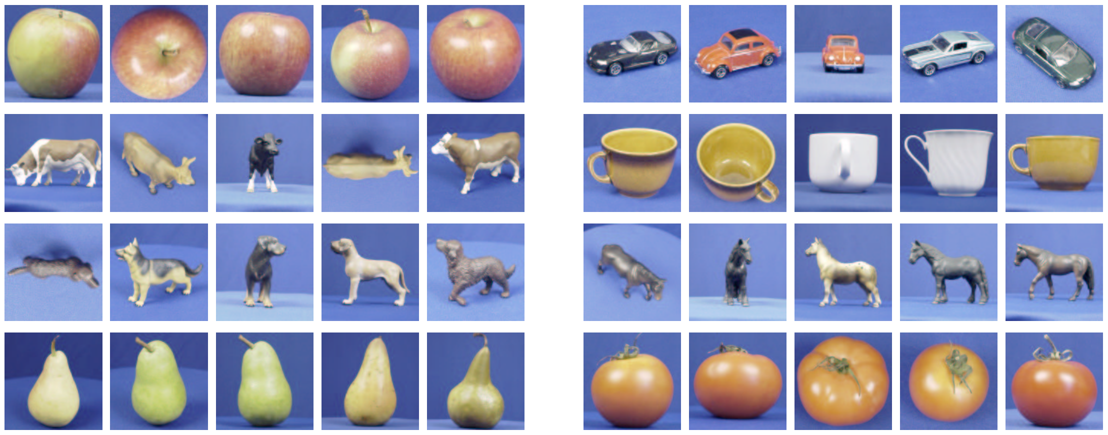


Figure 5: Example images from the ETH-80 objects database. Five images from each of the eight object classes (apple, cow, dog, pear, car, cup, horse, and tomato) are shown here.

in the left plot of part (a) due to distance clusters caused by drawing point sets from two distinct category models.

However, for the partial matching case (Figure 4 (b)), the L_1 approximation fails because it can handle only sets with equal cardinalities and requires all points to match to something. In contrast, the pyramid match can also approximate the partial matching for the unequal cardinality case: its matchings continue to follow the optimal matching's trend since it does not penalize outliers, as evidenced by the clustered points along the diagonal in the bottom right ranking quality plot. We have performed this same experiment using data generated from a uniform random point model, and the outcome was similar.

9. Discriminative Classification using Sets of Local Features

In this section we report on object recognition experiments using SVMs and provide baseline comparisons to other methods. We use the SVM implementation provided in the LIBSVM library (Chang and Lin, 2001) and train one-versus-all classifiers in order to perform multi-class classification.

Local affine- or scale-invariant feature descriptors extracted from a sparse set of interest points in an image have been shown to be an effective, compact representation (e.g., Lowe 2004; Mikolajczyk and Schmid 2001). This is a good context in which to test the pyramid match kernel, since such local features have no inherent ordering, and it is expected that the number of features will vary across examples. Given two sets of local image features, the pyramid match kernel (PMK) value reflects how well the image parts match under a one-to-one correspondence. Since the matching is partial, not all parts must have a match for the similarity to be strong.

In the following we experiment with two publicly available databases and demonstrate that our method achieves better or comparable object recognition performance at a significantly lower computational cost than other state-of-the-art approaches. All pyramid match run-times reported below include the time needed to compute both the pyramids and the weighted intersections, and were measured on 2.4 GHz processors with 2 GB of memory.

9.1 ETH-80 Data Set

A performance evaluation given by Eichhorn and Chapelle (2004) compares the set kernels developed by Kondor and Jebara (2003), Wolf and Shashua (2003), and Wallraven et al. (2003) in the context of an object categorization task using images from the publicly available ETH-80 database.[4] The experiment uses eight object classes, with 10 unique objects and five widely separated views of each, for a total of 400 images (see Figure 5). A Harris detector is used to find interest points in each image, and various local descriptors (SIFT features, Lowe 2004; JETs, Schmid and Mohr 1997; and raw image patches) are used to compose the feature sets. A one-versus-all SVM classifier is trained for each kernel type, and performance is measured via cross-validation, where all five views of an object are held out at once. Since no instances of a test object are ever present in the training set, this is a categorization task (as opposed to recognition of the same specific object).

The experiments show the polynomial-time "match kernel" (Wallraven et al., 2003) and Bhattacharyya kernel (Kondor and Jebara, 2003) performing best, with a classification rate of 74% using on average 40 SIFT features per image (Eichhorn and Chapelle, 2004). Using 120 interest points, the Bhattacharyya kernel achieves 85% accuracy. However, the study also concluded that the cubic complexity of the method made it impractical to use the desired number of features.

We evaluated the pyramid match kernel on this same subset of the ETH-80 database under the conditions provided in Eichhorn and Chapelle (2004), and it achieved a recognition rate of 83% using PCA-SIFT features (Ke and Sukthankar, 2004) from all Harris-detected interest points (averages 153 points per image) and $T = 8$. Restricting the input sets to an average of 40 interest points yields a recognition rate of 73%. Thus the pyramid match kernel (PMK) performs comparably to the others at their best for this data set, but is much more efficient than those tested above, requiring time only linear in the number of features.

In fact, the ability of a kernel to handle large numbers of features can be critical to its success. An interest operator may be tuned to select only the most salient features, but in our experiments we found that recognition rates always benefited from having larger numbers of features per image with which to judge similarity. The plots in Figure 6 depict the run-time versus recognition accuracy of the pyramid match kernel as compared to the match kernel (Wallraven et al., 2003), which has $O(dm^2)$ complexity. The match kernel computes kernel values by averaging over the distances between every point in one set and its nearest point in the other set. These are results from our own implementation of the match kernel. Each point in the figure represents one experiment, and both kernels were run with the exact same PCA-SIFT feature sets. The saliency threshold of the Harris interest operator was adjusted to generate varying numbers of features, thus trading off accuracy versus run-time. With no filtering of the Harris detector output (i.e., with a threshold of 0), there is on average 153 features per set. To extract an average of 256 features per set we used no interest operator and sampled features densely and uniformly from the images. Computing a kernel matrix for the same data with the two kernels yields nearly identical accuracy (left plot), but the pyramid match is significantly faster (center plot). Allowing the same amount of training time for both methods, the pyramid match produces much better recognition results (right plot).

9.2 Caltech-101 Data Set

We also tested our method with a challenging database of 101 object categories developed at Caltech, called the Caltech-101 (Fei-Fei et al., 2004). The creators of this database obtained the images

4. http://www.vision.ethz.ch/projects/categorization/.

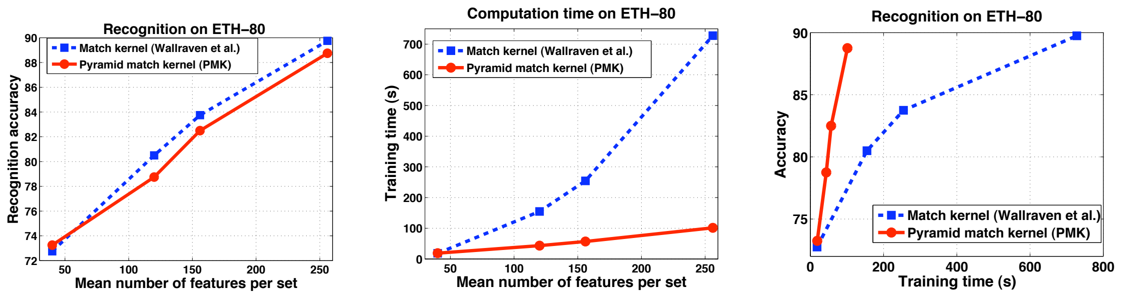


Figure 6: Both the pyramid match kernel and the match kernel of Wallraven et al. yield comparable recognition accuracy for input sets of the same size (left plot). Both benefit from using richer (larger) image descriptions. However, while the time required to learn object categories with the quadratic-time match kernel grows quickly with the input size, the time required by the linear-time pyramid match remains very efficient (center plot). Allowing the same run-time, the pyramid match kernel produces better recognition rates than the match kernel (right plot). Recognition accuracy is measured as the mean rate of correct predictions per class.

using Google Image Search, and many of the images contain a significant amount of intra-class appearance variation (see Figure 7). The Caltech-101 is currently the largest benchmark recognition data set (in terms of number of categories) available, and as such it has been the subject of many recent recognition experiments in the field. There are 8677 images in the data set, with between 31 to 800 images for each of the 101 categories.

For this data set, the pyramid match operated on sets of SIFT features projected to $d = 10$ dimensions using PCA, with each appearance descriptor concatenated with its corresponding positional feature (image position normalized by image dimensions). The features were extracted on a uniform grid at every 8 pixels in the images, that is, no interest operator was applied. The regions extracted for the SIFT descriptor were about 16 pixels in diameter, and each set had on average $m = 1140$ features. We trained the algorithm using unsegmented images. Since the pyramid match seeks a strong correspondence with some subset of the images' features, it explicitly accounts for unsegmented, cluttered data. Classification was again done with a one-vs-all SVM, and we set the number of grid shifts $T = 1$, and summed over kernels corresponding to pyramids with finest side lengths of 5,7, and 9. Note that some images were rotated by the creators of the database, causing some triangular artifacts in the corners of some images; this is how the images are provided to any user.

As a baseline, we also experimented with the bag-of-words representation and an RBF kernel: $K(\mathbf{x},\mathbf{y}) = \exp\left(-||\mathbf{x}-\mathbf{y}||/\sigma\right)$ for bag-of-words histograms $\mathbf{x}$ and $\mathbf{y}$. We used hierarchical k-means to cluster a corpus of two million example features from the Caltech-101 data set to form each vocabulary, and we experimented with both L_2 and the Mahalanobis distance as the basis for the clustering; classification results were similar with either distance. Generating a vocabulary took about five hours, and computing a kernel matrix required about one half hour. We experimented with a large number of parameters, and the dotted red line in Figure 8 (a) reflects the very best results we could obtain over any parameter setting for this technique. Specifically, the number of

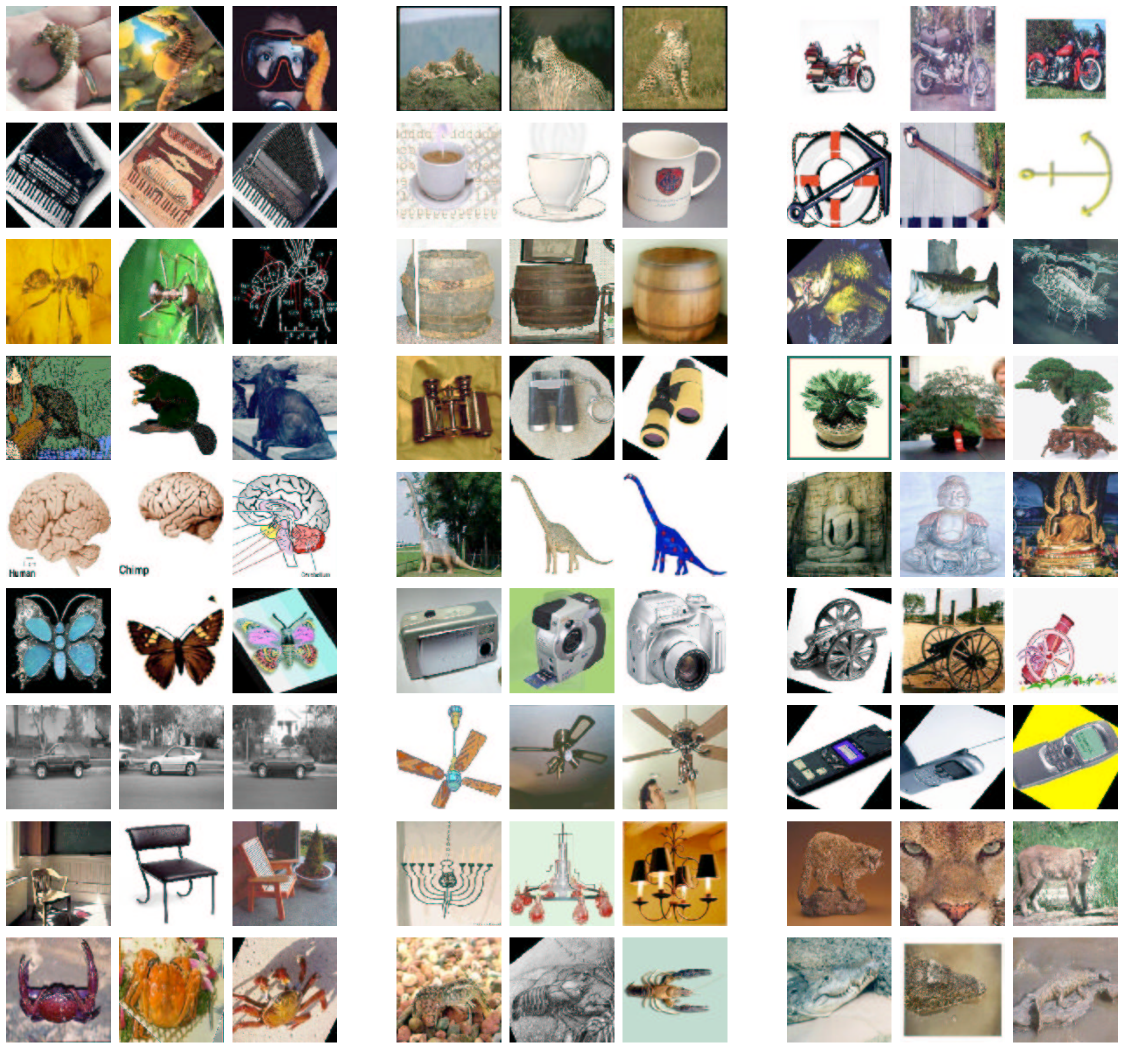


Figure 7: Example images from the Caltech-101 database. Three images are shown for each of 27 of the 101 categories.

feature prototypes (“words”) was tested at values ranging from 1000 to 100,000 (2200 was best); the kernel’s σ parameter was tested from values of 500 to 10,000 (1000 was best).

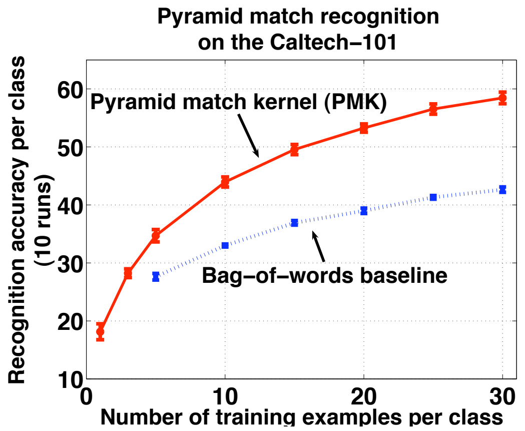
Figure 8 (a) shows the multi-class category recognition results using the PMK (red solid line) alongside the corresponding results for a bag-of-words baseline approach (blue dotted line). The recognition scores are given for varying numbers of training set sizes, ranging from one to 30 examples per class. For each training set size, the mean and standard deviation of the pyramid match accuracy over 10 runs is shown, where for each run we randomly select the training examples and use all the remaining database images as test examples. All recognition rates have been normalized according to the number of novel test images per class; that is, the mean recognition rate per class is the average normalized score for all 101 categories. Using only one training example per class, the pyramid match achieves an average recognition rate per class of 18%. Note that chance performance with any number of training examples would be just 1%. Using 15 examples (a standard point of comparison), the pyramid match achieves an average recognition rate per class of 50%. Experiments comparing the recognition accuracy of the pyramid match kernel to an optimal partial matching kernel reveal that very little loss in accuracy is traded for speedup factors of several orders of magnitude (Grauman, 2006, , pages 118-121).

Over all the parameter configurations we tested, the best performance we could achieve with the bag-of-words on this data set is substantially poorer than that of the pyramid match, as seen in Figure 8 (a). In addition, the bag-of-word approach's accuracy was fairly sensitive to the parameter settings; for instance, over all the parameters we tested, the mean accuracy for 15 training examples per class varied from 28% to 37%. This indicates the difficulty of establishing an optimal flat quantization of the feature space for recognition, and also suggests that the partial matching ability of the pyramid match is beneficial for tolerating outlier features without skewing the matching cost.

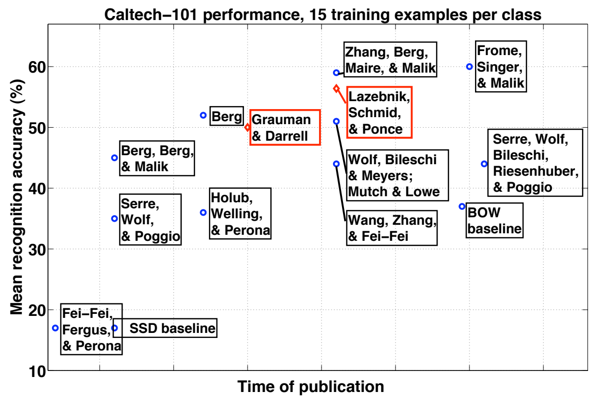
Figure 8 (b) shows all published results on this data set as a function of time since the data was released, including the PMK and results published more recently by other authors (Holub et al. 2005b; Serre et al. 2007; Wolf et al. 2006; Wang et al. 2006; Berg et al. 2005; Berg 2005; Fei-Fei et al. 2004; Mutch and Lowe 2006; Lazebnik et al. 2006; Zhang et al. 2006; Frome et al. 2007). From this comparison we can see that even with its extreme computational efficiency, the pyramid match kernel achieves results that are competitive with the state-of-the-art. We have previously obtained 50% accuracy on average (0.9% standard deviation) when using the standard 15 training examples per category (Grauman and Darrell, 2006). In addition, the PMK with sets of spatial features yields very good accuracy on this data set: 56.4% for 15 training examples per class (Lazebnik et al., 2006). These results are based on a special case of the pyramid match, where multiple pyramid match kernels computed on spatial features are summed over a number of quantized appearance feature channels. This is among the very best accuracy rates reported to-date on the data set, and about three percentage points below the most accurate result of 60% obtained recently by Frome et al. (2007), which compares images using a discriminative local distance function.

Figure 8 (c) shows the recognition accuracy of various methods as a function of computation time, measured in terms of the time required to train a classifier (left plot) and the time required to classify a novel example (right plot). We collected these time estimates directly from the authors, and every response we received is plotted here. (So not every method represented in (b) is represented in (c) due to a lack of information on computation times.) For both plots in part (c), computation time is measured *excluding* the time required to extract image features for all methods. For display purposes, in the left plot, we plot the square root of the training time; nearest-neighbor classifiers such as the method of Berg et al. require no training time beyond feature extraction. In the right plot, we plot the log of the classification time estimates to reflect the relative complexities in terms of orders of magnitude. This figure illustrates the PMK's clear practical cost advantages. Pyramid matching any two examples requires just 0.0001 s on this data set, and classifying a novel example requires a fraction of a second; in contrast, many other techniques require on the order of a minute to classify a single example, and yield varying accuracies relative to the PMK.

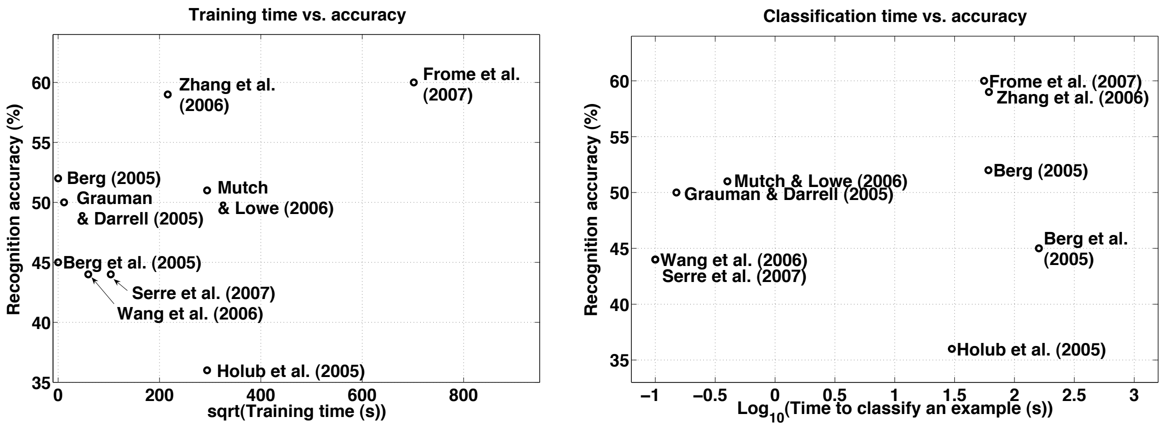
Pyramid match comparisons require only $O(mL)$ time for the result of 50% accuracy. For the spatial pyramid match result of 56.4% accuracy obtained by Lazebnik et al. (2006) there is an additional one-time $O(mk)$ cost of assigning each of a set's features to one of the k pre-established quantized appearance features; however, once a point set has been mapped to its prototypes, they need not be re-computed to match against additional sets.



(a) PMK vs. BOW categorization results



(b) Comparison of all published results



(c) Accuracy vs. time complexity

Figure 8: Recognition results on the Caltech-101 data set. Plot (a) shows the mean and standard deviation for the recognition accuracy using the pyramid match kernel (red solid line) and a bag-of-words baseline (blue dotted line) when given varying numbers of training examples per class, over 10 runs with randomly selected training examples. These are recognition rates that have been normalized according to the number of test examples per class. Plot (b) shows all published results on the same Caltech-101 data set over time since the database's release. Each recognition rate on this plot represents the mean accuracy per class when using 15 training examples per class. Grauman & Darrell refers to our PMK results using appearance descriptors, and Lazebnik et al. report results using a PMK applied to 2D spatial features in each channel of quantized appearance. The two plots in (c) show the accuracy of various methods as a function of their computational costs, in terms of average training (left, square root scale) and testing times (right, log scale), again when each method uses 15 training examples per class. All times exclude the cost of computing the various image features employed.

10. Learning a Function over Sets of Features

In the following experiments we demonstrate the pyramid match applied to two regression problems: time of publication inference from a collection of research papers, and articulated pose estimation from monocular silhouettes. We again use the SVR implementation of Chang and Lin (2001). In these experiments we use an ε-insensitive loss function in order to obtain solutions that are defined in terms of a small subset of the training examples, and which provide good generalization ability. For all experiments, the SVR parameters C and ε were chosen using cross-validation.

10.1 Estimating a Document's Time of Publication

We have applied our method to learn a function that maps a bag of local latent semantic features extracted from a research paper (written in a specific field) to an estimate of the paper's year of publication. While much work has been done using bag-of-words representations and latent semantic concept models for text classification and information retrieval, we are not aware of previous work showing *bags of local semantic features*, nor demonstrating direct regression on documents to estimate their publication times.

The bag-of-words model is a widely used representation in text processing in which each document is represented by a vector giving the frequencies of words that occur in it, and it has been used in kernel methods (Shawe-Taylor and Cristianini, 2004). The well-known limitation of such a model, however, is that it encodes nothing about the semantic relationships between words. Each word is treated as an isolated entity, ignoring the fact that distinct words may share degrees of semantic equivalency (e.g., synonyms), or may have different connotations in different contexts (e.g., homonymy). Researchers have therefore adopted latent semantic indexing (LSI) (Deerwester et al., 1990) to instill semantic cues into the basic bag-of-words representation. LSI projects data onto a subspace determined using singular value decomposition (SVD) to represent document-term frequency vectors in a lower-dimensional space where semantic relations are better preserved. Generally the subspace for LSI is learned from document vectors that give the frequency of occurrence for each given word (e.g., Cristianini et al., 2002), which means each document is mapped to a point in a "semantic space" where documents with correlated word frequencies are related.

However, requiring a document to map to a single point in this space assumes that inputs have no clutter (e.g., extra words caused from OCR errors, or text inserted from a webpage ad), and that each document can be represented as a linear combination of the document-level concepts recovered by LSI. Instead, we propose treating documents as bags of *word meanings* by learning the subspace for LSI from *term* vectors, which record the frequency with which a word occurs in each given document in a training corpus. Each document is then a bag of local semantic features, and two documents are compared with the partial matching implied by the pyramid match kernel, that is, in terms of how well (some subset) of the LSI term-space projections can be put into correspondence. Note that standard kernels (e.g., linear, RBF, polynomial) cannot be used with the bag of word meanings representation, since it represents each word with a real-valued vector. Additionally, our method makes it possible to learn a latent semantic space from narrower contexts than entire documents (e.g., paragraphs or sentences) and then represent documents by their component features in this space.

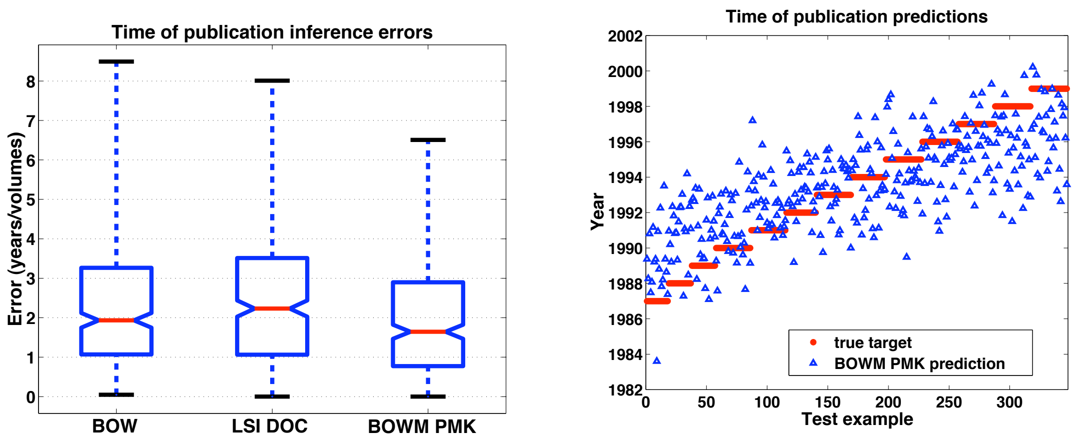


Figure 9: Inferring the time of publication for papers from 13 volumes of NIPS proceedings. Boxplots (left) compare errors produced by three methods with a support vector regressor: bag-of-words (BOW) and latent semantic document-vectors (LSI DOC) with linear kernels, and "bag of word meanings" with the pyramid match kernel (BOWM PMK). Lines in center of boxes denote median value, top and bottom of boxes denote upper and lower quartile values, dashed lines show the extent of the rest of the errors. The plot on the right shows the true targets and corresponding predictions made by the pyramid match method (BOWM PMK).

We have experimented with a database containing 13 volumes of Neural Information Processing Systems (NIPS) proceedings—a total of 1,740 documents, available online.[5] For each paper, we extracted the text from the abstract up to (but not including) the references section. While authors' names and bibliography information would likely be good indicators of publication time, they were excluded from the bags of features because we want our method to learn a function indicating topic trends over time, as opposed to a look-up table of dates and names. We applied standard steps to pre-process the text data. Suffixes were removed from words using the Porter stemmer (Porter, 1980), and the WordNet set of stop-list words were removed. Finally, co-occurrence matrices were weighted using term frequency-inverse document frequency (*tf-idf*) to normalize for text lengths and distill the most informative words.

Figure 9 shows results for regressing directly on the year of publication for a NIPS paper using the classic bag-of-words approach (BOW), a standard approach applying LSI at the document level (LSI DOC) (Cristianini et al., 2002), and our method with bags of word meanings (BOWM PMK). All methods were trained with the same randomly selected subset of the data (1393 examples) and tested with the remaining 347 examples. Our approach with bags of word meanings performs the best, with a median error of 1.6 years. The PMK values took on average 0.002 seconds to compute. Using a paired difference T-test with $\alpha = 0.01$, we found the difference in performance between our approach and the two existing methods to be statistically significant.

5. http://www.cs.toronto.edu/ roweis/data.html.

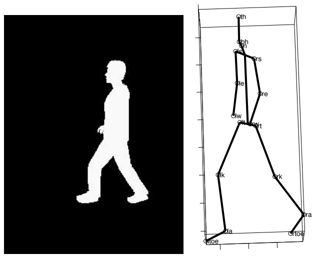


Figure 10: A training example generated with graphics software is composed of a silhouette and its corresponding 3-D pose, as represented by the 3-D positions of 15 joint positions.

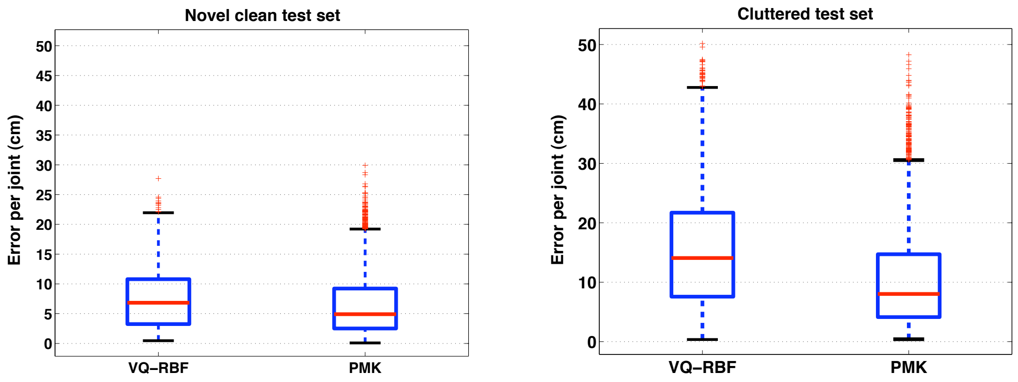
10.2 Inferring 3-D Pose from Shape Features

In this set of experiments, we use regression with the pyramid match kernel to directly learn the mapping between monocular silhouette images of humans and the corresponding articulated 3-D body pose. Many vision researchers have addressed the difficult problem of articulated pose estimation; recent approaches have attempted to directly learn the relationship between images and 3-D pose (Agarwal and Triggs, 2004; Shakhnarovich et al., 2003; Grauman et al., 2003). Like these techniques, we learn a function that maps observable image features to 3-D poses.

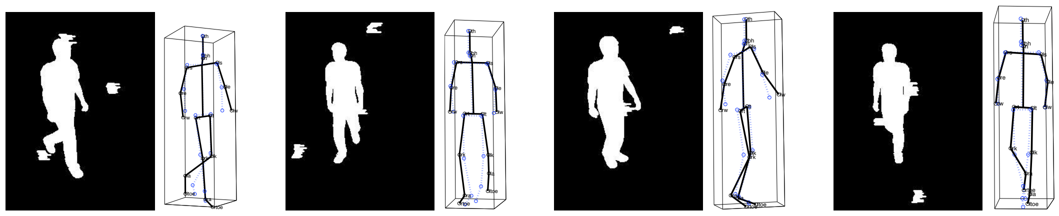
However, whereas ordered, fixed-length feature sets are required in the methods of Grauman et al. (2003) and Shakhnarovich et al. (2003) (i.e., points are extracted in sequence around the contour, or features are taken from fixed image windows), our method accepts unordered features and inputs that may vary in size. This is a critical difference: images will naturally have varying numbers of features (due to occlusions, clutter, translations, shape deformations, viewpoint changes, etc.), and a robust global ordering among features within a single view may not be possible in the presence of viewpoint and pose variations. In the pose estimation method of Agarwal and Triggs (2004) a bag-of-words representation is used: local features are mapped to pre-established prototypes, and every image is represented by a frequency vector counting the number of times each prototype occurred. A relevance vector machine is then trained using these vectors with a Gaussian kernel. While this representation allows unordered features, it can be sensitive to clutter, as we will show below.

As a training set, we used 3,040 images of realistic synthetic images of pedestrians generated using the graphics package POSER. Each image was rendered from a humanoid model with randomly adjusted anatomical shape parameters walking in a randomly selected direction. For each image, both the silhouette and the 3-D locations of 15 landmarks on the model's skeleton corresponding to selected anatomical joints were recorded (see Figure 10). Regressors are trained with silhouette inputs and produce 3-D joint position outputs. Once regressors have been trained with the synthetic data, we can use them to perform regression with either additional synthetic examples (for which we have ground truth poses), or with real image data (for which we can only evaluate predicted poses qualitatively).

We represent each silhouette with a set of local contour descriptors. At each contour point, we extract a shape context histogram (Belongie et al., 2002), which bins nearby edge pixels into a log-polar histogram, thereby encoding local shape. For each shape context histogram, we used 12 angular and five radial bins with a fixed scale to capture only local points relative to each edge point. To form a more compact descriptor, we used low-dimensional PCA projections ($d = 5$) of



(a) Pose estimation errors



(b) Noisy synthetic test examples

Figure 11: Pose inference results. The top row (a) gives a quantitative evaluation of performance on synthetic data with ground truth. The boxplots compare the error distributions for the pyramid match kernel (PMK) and an RBF kernel over prototype frequency vectors (VQ-RBF). Errors are measured by the distance between the 15 true and inferred joint positions. Results for two test sets are shown: a set of novel, clean silhouettes (left plot), and a set with randomly generated clutter or extra foreground blobs (right plot). The bottom row (b) shows example poses inferred by our method from synthetic cluttered silhouettes. In each, the true pose (solid black) is overlayed with the estimate (dotted blue). These examples contain average case errors.

the initial 60-D shape context histogram features. Thus each silhouette shape is represented by an unordered set of shape context subspace features, and each set varies in size due to the varying number of contour points per image. Note that although this representation does not contain explicit spatial constraints, the overlap between neighboring shape context histograms provides an implicit encoding of how the features should be related spatially.

The number of contour points (and thus features) per image varied from $m = 405$ to $m = 878$. The multi-resolution histograms used by the pyramid match contained ten resolution levels, as determined by the diameter of the features in the training examples, and each contained on average 2644 non-empty bins. Computing a PMK value required about 0.002 seconds. For each dimension of the pose targets, we trained an ε-insensitive SVR using the pyramid match kernel matrix between the training examples. Each SVR had on average 308 support vectors.

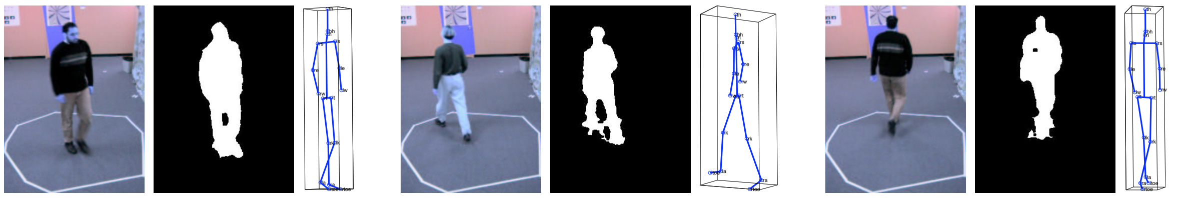


Figure 12: Pose inference on real images.

As a baseline, we also implemented a method that uses frequency vectors over feature prototypes to represent each image (using Agarwal and Triggs, 2004, as a guideline). To obtain bag-of-words histograms, vector quantization (VQ) is performed on the shape context subspace features found in the training set to establish a set of 100 prototype features. Then all of the features detected in a new image are mapped to a 1-D frequency histogram over these prototypes using soft voting with Gaussian weights. A Gaussian RBF kernel is then applied, with γ chosen based on the maximum inter-feature distance. In the following we will refer to this baseline as VQ-RBF.

For a novel test set of 300 POSER-generated silhouettes, the pose inferred by our method had a median error of 4.9 cm per joint position. For the same test set, VQ-RBF obtained a slightly worse median error of 6.8 cm (see Figure 11). Using a paired difference T-test with $\alpha = 0.001$, we found the difference in performance between the two methods to be statistically significant.

The silhouette contours produced with POSER are of course very “clean”, that is, the shapes are perfectly segmented since they were formed by directly projecting the CG body. While it is reasonable to train a model with this well-segmented data, we can expect real-world examples to exhibit poorer foreground-background segmentations due to occlusions, clutter, shadows, or backgrounds that look similar to the foreground object. Therefore, we altered the silhouettes for a separate test set of 300 examples to introduce clutter and occlusions; starting with a POSER silhouette, we generated one to eight random blob regions in the image for which the foreground/background labels were swapped. The blobs’ positions, sizes, and shapes were generated randomly. The result is a test set that attempts to mimic real-world occlusions and clutter, yielding imperfect contours for which estimating pose is more challenging (see Figure 11).

On the cluttered test set, our method inferred poses with a median error of 8.0 cm per joint, while VQ-RBF had a median error of 14.1 cm (see Figure 11). Again, using a paired difference T-test, we found the difference in performance to be statistically significant: with 99.99% confidence, the pyramid match yields average errors that are smaller than those of VQ-RBF by amounts between 4.5 and 5.2 cm per joint.

This experiment demonstrates the pyramid match kernel’s robustness to superfluous features in an input. The blobs added to the cluttered test set introduced extra contour features to the silhouettes, and they altered parts of the true contour in cases where they overlapped with the real silhouette. The VQ-RBF distance over prototype frequency vectors (bags-of-words) essentially penalizes any features that have no “match” in a support vector training example’s set of features. In contrast, the pyramid match’s partial matching rewards similarity only for the features placed into correspondence, and ignores extraneous clutter features. This is an important advantage for many vision applications, where segmentation errors, viewpoint changes, or image noise will often affect which features (and how many) are detected.

Finally, we applied our method to a test set of real images of various subjects walking through a scene. A basic background subtraction method was used, which resulted in rough segmentations; body parts are frequently truncated in the silhouettes where the background is not highly textured, or else parts are inaccurately distended due to common segmentation problems from shadows. Ground truth poses are not available for this test set, but Figure 12 shows some example output poses inferred by our method.

11. Conclusions

We have developed a new efficient kernel function that handles unordered sets of features, and we have shown its suitability for both discriminative classification and regression. The pyramid match kernel approximates the optimal partial matching by computing a weighted intersection over multi-resolution histograms, and it requires time only linear in the number of features. The kernel is robust to clutter since it does not penalize the presence of extra features, it respects the co-occurrence statistics inherent in the input sets, and it is provably positive-definite. We have applied our kernel to a variety of tasks—object recognition, pose estimation, and time of publication inference—and have demonstrated our method's advantages over other state-of-the-art techniques. Source code for the pyramid match kernel is available at `http://people.csail.mit.edu/jjl/libpmk/`.

Recently we have developed a method to generate data-dependent pyramid partitions, which yield more accurate matching results in high-dimensional feature spaces (Grauman and Darrell, 2007). In ongoing work we are considering alternative weighting schemes on the bins for the pyramid match, and developing methods for sub-linear time approximate similarity search over partial correspondences.

Acknowledgments

We would like to thank Piotr Indyk, Regina Barzilay, Michael Collins, and John Lee for helpful discussions, and Michael Collins for suggesting the application of estimating publication times. We also thank John Lee for sharing his bag-of-words results on the Caltech-101 database, and the JMLR reviewers for their helpful comments.

References

A. Agarwal and B. Triggs. 3D human pose from silhouettes by relevance vector regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Washington, DC, June 2004.

P. Anandan. *Measuring Visual Motion from Image Sequences*. PhD thesis, University of Massachusetts, Amherst, 1987.

S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(24):509–522, April 2002.

A. Berg. *Shape Matching and Object Recognition*. PhD thesis, U.C. Berkeley, Computer Science Division, Berkeley, CA, December 2005.

A. Berg, T. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, San Diego, CA, June 2005.

S. Boughhorbel, J-P. Tarel, and F. Fleuret. Non-Mercer kernels for SVM object recognition. In *British Machine Vision Conference*, London, UK, September 2004.

S. Boughorbel, J. Tarel, and N. Boujemaa. The intermediate matching kernel for image local features. In *International Joint Conference on Neural Networks*, Montreal, Canada, August 2005.

P. Burt and E. Adelson. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, COM-31,4:532–540, 1983.

C. Chang and C. Lin. *LIBSVM: a library for SVMs*, 2001.

O. Chapelle, P. Haffner, and V. Vapnik. SVMs for histogram-based image classification. *Transactions on Neural Networks*, 10(5), September 1999.

T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

N. Cristianini, J. Shawe-Taylor, and H. Lodhi. Latent semantic kernels. *Journal of Intelligent Information Systems*, 18(2/3):127–152, 2002.

G. Csurka, C. Bray, C. Dance, and L. Fan. Visual categorization with bags of keypoints. In *Proceedings of European Conference on Computer Vision*, Prague, Czech Republic, May 2004.

M. Cuturi and J-P. Vert. Semigroup kernels on finite sets. In *Advances in Neural Information Processing*, Vancouver, Canada, December 2005.

S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

J. Eichhorn and O. Chapelle. Object categorization with SVM: kernels for local features. Technical report, MPI for Biological Cybernetics, 2004.

L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object cateories. In *Workshop on Generative Model Based Vision*, Washington, D.C., June 2004.

R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Madison, WI, 2003.

A. Frome, Y. Singer, and J. Malik. Image retrieval and classification using local distance functions. In B. Scholkopf, J.C. Platt, and T. Hofmann, editors, *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

T. Gartner. A survey of kernels for structured data. *Multi Relational Data Mining*, 5(1):49–58, July 2003.

K. Grauman. *Matching Sets of Features for Efficient Retrieval and Recognition*. PhD thesis, MIT, 2006.

K. Grauman and T. Darrell. Fast contour matching using approximate Earth Mover's Distance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Washington D.C., June 2004.

K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proceedings IEEE International Conference on Computer Vision*, Beijing, China, October 2005.

K. Grauman and T. Darrell. Pyramid match kernels: Discriminative classification with sets of image features. Technical Report MIT-CSAIL-TR-2006-020, MIT CSAIL, Cambridge, MA, March 2006.

K. Grauman and T. Darrell. Approximate correspondences in high dimensions. In B. Scholkopf, J.C. Platt, and T. Hofmann, editors, *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

K. Grauman, G. Shakhnarovich, and T. Darrell. Inferring 3D structure with a statistical image-based shape model. In *Proceedings IEEE International Conference on Computer Vision*, Nice, France, October 2003.

E. Hadjidemetriou, M. Grossberg, and S. Nayar. Multiresolution histograms and their use for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(7):831–847, July 2004.

E. Hayman, B. Caputo, M. Fritz, and J.-O. Eklundh. On the significance of real-world conditions for material classification. In *Proceedings of European Conference on Computer Vision*, Prague, Czech Republic, 2004.

A. Holub, M. Welling, and P. Perona. Combining generative models and fisher kernels for object recognition. In *Proceedings IEEE International Conference on Computer Vision*, Beijing, China, October 2005a.

A. Holub, M. Welling, and P. Perona. Exploiting unlabelled data for hybrid object classification. Technical report, NIPS 2005 Workshop on Inter-Class Transfer, Vancouver, Canada, December 2005b.

P. Indyk and N. Thaper. Fast image retrieval via embeddings. In *3rd International Workshop on Statistical and Computational Theories of Vision*, Nice, France, Oct 2003.

T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing*, Denver, CO, December 1999.

Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Washington, D.C., June 2004.

R. Kondor and T. Jebara. A kernel between sets of vectors. In *Proceedings of the International Conference on Machine Learning*, Washington, D.C., August 2003.

J. Lafferty and G. Lebanon. Information diffusion kernels. In *Advances in Neural Information Processing*, Vancouver, Canada, December 2002.

S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, New York City, NY, June 2006.

T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43(1):29–44, 2001.

D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, January 2004.

S. Lyu. Mercer kernels for object recognition with local features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, San Diego, CA, June 2005.

J. Malik and P. Perona. Preattentive texture discrimination with early vision mechanisms. *Journal of Optical Society of America*, 7(5):923–932, May 1990.

K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *Proceedings IEEE International Conference on Computer Vision*, Vancouver, Canada, July 2001.

P. Moreno, P. Ho, and N. Vasconcelos. A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. In *Advances in Neural Information Processing*, Vancouver, December 2003.

J. Mutch and D. Lowe. Multiclass object recognition using sparse, localized features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, New York City, NY, June 2006.

F. Odone, A. Barla, and A. Verri. Building kernels from binary strings for image matching. *IEEE Transactions on Image Processing*, 14(2):169–180, February 2005.

M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

C. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

D. Roobaert and M. Van Hulle. View-based 3D object recognition with support vector machines. In *IEEE International Workshop on Neural Networks for Signal Processing*, Madison, WI, Aug 1999.

Y. Rubner, C. Tomasi, and L. Guibas. The Earth Mover's Distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

C. Schmid and R. Mohr. Local greyvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–534, 1997.

T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio. Object recognition with cortex-like mechanisms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):411–426, 2007.

F. Shaffalitzky and A. Zisserman. Automated scene matching in movies. In *Proceedings, Challenge of Image and Video Retrieval*, London, U.K., July 2002.

G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. In *Proceedings IEEE International Conference on Computer Vision*, Nice, France, October 2003.

Y. Shan, S. Sawhney, and R. Kumar. Vehicle identification between non-overlapping cameras without direct feature matching. In *Proceedings IEEE International Conference on Computer Vision*, 2005.

A. Shashua and T. Hazan. Algebraic set kernels with application to inference over local image representations. In *Advances in Neural Information Processing*, Vancouver, Canada, Dec 2005.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering object categories in image collections. In *Proceedings IEEE International Conference on Computer Vision*, Beijing, China, October 2005.

J. Sivic and A. Zisserman. Video Google: a text retrieval approach to object matching in videos. In *Proceedings IEEE International Conference on Computer Vision*, Nice, France, Oct 2003.

M. Swain and D. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

T. Tuytelaars and L. Van Gool. Content-based image retrieval based on local affinely invariant regions. In *3rd Intl Conference on Visual Information Systems*, Amsterdam, the Netherlands, June 1999.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.

C. Wallraven, B. Caputo, and A. Graf. Recognition with local features: the kernel recipe. In *Proceedings IEEE International Conference on Computer Vision*, Nice, France, October 2003.

G. Wang, Y. Zhang, and L. Fei-Fei. Using dependent regions for object categorization in a generative framework. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, New York City, NY, June 2006.

J. Weston, B. Scholkopf, E. Eskin, C. Leslie, and W. Noble. Dealing with large diagonals in kernel matrices. In *Principles of Data Mining and Knowledge Discovery*, volume 243 of *SLNCS*, 2002.

J. Willamowski, D. Arregui, G. Csurka, C. R. Dance, and L. Fan. Categorizing nine visual classes using local appearance descriptors. In *ICPR Workshop on Learning for Adaptable Visual Systems*, 2004.

L. Wolf, S. Bileschi, and E. Meyers. Perception strategies in hierarchical vision systems. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, New York City, NY, June 2006.

L. Wolf and A. Shashua. Learning over sets using kernel principal angles. *Journal of Machine Learning Research*, 4:913–931, Dec 2003.

H. Zhang, A. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, New York City, NY, June 2006.

H. Zhang and J. Malik. Learning a discriminative classifier using shape context distances. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Madison, WI, June 2003.

Journal of Machine Learning Research 8 (2007) 761-773 Submitted 9/06; Revised 12/06; Published 4/07

Infinitely Imbalanced Logistic Regression

Art B. Owen OWEN@STAT.STANFORD.EDU
Department of Statistics
Stanford University
Stanford CA, 94305, USA

Editor: Yi Lin

Abstract

In binary classification problems it is common for the two classes to be imbalanced: one case is very rare compared to the other. In this paper we consider the infinitely imbalanced case where one class has a finite sample size and the other class's sample size grows without bound. For logistic regression, the infinitely imbalanced case often has a useful solution. Under mild conditions, the intercept diverges as expected, but the rest of the coefficient vector approaches a non trivial and useful limit. That limit can be expressed in terms of exponential tilting and is the minimum of a convex objective function. The limiting form of logistic regression suggests a computational shortcut for fraud detection problems.

Keywords: classification, drug discovery, fraud detection, rare events, unbalanced data

1. Introduction

In many applications of logistic regression one of the two classes is extremely rare. In political science, the occurrence of wars, coups, vetos and the decisions of citizens to run for office have been modelled as rare events; see King and Zeng (2001). Bolton and Hand (2002) consider fraud detection, and Zhu et al. (2005) look at drug discovery. In other examples the rare event might correspond to people with a rare disease, customer conversions at an e-commerce web site, or false positives among a set of emails marked as spam.

We will let $Y \in \{0,1\}$ denote a random response with the observed value of Y being $y=1$ in the rare case and $y=0$ in the common case. We will suppose that the number of observations with $y=0$ is so large that we have a satisfactory representation of the distribution of predictors in that setting. Then we explore the limit as the number of $y=0$ cases tends to infinity while the number of observed cases with $y=1$ remains fixed. It is no surprise that the intercept term in the logistic regression typically tends to $-\infty$ in this limit. The other coefficients can however tend to a useful limit.

The main result (Theorem 8 below) is that under reasonable conditions, the intercept term tends to $-\infty$ like $-\log(N)$ plus a constant, while the limiting logistic regression coefficient $\beta=\beta(N)$ satisfies

$$\bar{x} = \frac{\int e^{x'\beta} x \, dF_0(x)}{\int e^{x'\beta} \, dF_0(x)} \tag{1}$$

where F_0 is the distribution of X given $Y = 0$ and $\bar{x}$ is the average of the sample x_i values for which $y = 1$. The limiting solution is the exponential tilt required to bring the population mean of X given $Y = 0$ onto the sample mean of X given $Y = 1$.

When F_0 is the $N(\mu_0, \Sigma_0)$ distribution for finite nonsingular Σ_0 then

$$\lim_{N\to\infty} \beta(N) = \Sigma_0^{-1}(\bar{x} - \mu_0). \tag{2}$$

Equation (2) reminds one of a well known derivation for logistic regression. If the conditional distribution of X given that $Y = y$ is $N(\mu_y, \Sigma)$ then the coefficient of X in logistic regression is $\Sigma^{-1}(\mu_1 - \mu_0)$. Equation (2) however holds without assuming that the covariance of X is the same for $Y = 0$ and $Y = 1$, or even that X is Gaussian given that $Y = 1$.

The outline of the paper is as follows. Section 2 gives three numerical examples that illustrate the limiting behavior of β. One is a positive result in which we see β approaching the value computed from (2). The other two examples are negative results where (1) does not hold. Each negative case illustrates the failure of an assumption of Theorem 8. In one case there is no nontrivial estimate of β at any $N > 0$ while in the other β diverges as $N \to \infty$. Section 3 formally introduces the notation of this paper. It outlines the results of Silvapulle (1981) who completely characterizes the conditions under which unique logistic regression estimates exist in the finite sample case. The infinite sample case differs importantly and requires further conditions. A stronger overlap condition is needed between the two X distributions. Also, the distribution of X given $Y = 0$ must not have tails that are too heavy, an issue that cannot arise in finite samples from $\mathbb{R}^d$. Section 4 proves the results in this paper. A surprising consequence of Equation (1) is that the x values when $y = 1$ only appear through their average $\bar{x}$. Section 5 shows that for the drug discovery example of Zhu et al. (2005), we can replace all data points with $y = 1$ by a single one at $(\bar{x}, 1)$ with minimal effect on the estimated coefficient, apart from the intercept term. Section 6 discusses how these results can be used in deciding which unlabelled data points to label, and it shows how the infinitely imbalanced setting may lead to computational savings.

We conclude this introduction by relating the present work to the literature on imbalanced data. The English word "unbalanced" seems to be more popular, at least on web pages, than is "imbalanced". But the latter term has been adopted for this special setting in two recent workshops: AAAI 2000 and ICML 2003, respectively Japkowicz (2000) and Chawla et al. (2003). An extensive survey of the area is given by Chawla et al. (2004). In that literature much attention is paid to undersampling methods in which some of the available cases with $Y = 0$ are either randomly or strategically removed to alleviate the imbalance. Another approach is oversampling in which additional, possibly synthetic, cases are generated with $Y = 1$. It is also clear that prediction accuracy will be very good for a trivial method that always predicts $y = 0$ and so one needs to take care about misclassification cost ratios and prior probability ratios.

2. Numerical Examples

For illustration, suppose that when $Y = 0$ that $X \sim N(0, 1)$ and that we have one single observation with $y = 1$ and it has $x = 1$. To study this case we use logistic regression on $(x_i, y_i) = (\Phi^{-1}((i - 1/2)/N), 0)$ for $i = 1, \ldots, N$ and $(x_{N+1}, y_{N+1}) = (1, 1)$. Here Φ is the cumulative distribution function (CDF) of the $N(0, 1)$ distribution. As N increases the problem becomes more imbalanced and the N points used produce an ever better approximation to the normal distribution. Taking stratified X_i reduces inessential variation in the computation making the convergence pattern

N	α	Ne^{α}	β
10	-3.19	0.4126	1.5746
100	-5.15	0.5787	1.0706
1,000	-7.42	0.6019	1.0108
10,000	-9.71	0.6058	1.0017
100,000	-12.01	0.6064	1.0003

Table 1: Logistic regression intercept α and coefficient β for imbalanced data described in the text. There are N observations with $Y = 0$ and stratified $X \sim N(0,1)$ and one observation with $Y = 1$ and $X = 1$.

clearer. Some resulting values are shown in Table 1. From this table it seems clear that as $N \to \infty$, the intercept term is diverging like $-\log(N)$ while the coefficient of X is approaching the value 1 that we would get from Equation (2). Theorem 8 below shows that such is indeed the limiting behavior.

Next we repeat the computation replacing Φ by the CDF of the standard Cauchy distribution with density $1/(\pi(1+x^2))$. The results are shown in Table 2. Here it is clear that $\beta \to 0$ as $N \to \infty$ and α appears to behave like a constant minus $\log(N)$. It is not surprising that $\beta \to 0$ in this limit. The Cauchy distribution has tails far heavier than the logistic distribution. If $\beta \neq 0$ then the log likelihood (4) that we introduce in Section 3 is $-\infty$. The likelihood is maximized at $\beta = 0$ and $\alpha = -\log(N+1)$. We get slightly different values in Table 2 because the uniform distribution over N Cauchy quantiles that we use has lighter tails than the actual Cauchy distribution it approaches. The heavy tails of the Cauchy distribution make it fail a condition of Theorem 8. The finite sample setting does not need a tail condition on the distribution of X, beyond an assumption that all observed values are finite.

In the next example we use the $U(0,1)$ distribution for X given $Y = 0$. This time we use $n = 2$ points with $y = 1$. One has $x = 1/2$ and the other has $x = 2$. The results are shown in Table 3. Once again the value β does not appear to be converging to a limit. It cannot be due to heavy tails, because the $U(0,1)$ distribution has bounded support. On further thought, we see that $\bar{x} = 5/4$. There is no possible way for an exponential tilt like (1) to reweight the $U(0,1)$ distribution to have mean $5/4$. This example also fails one of the conditions of Theorem 8. We need the point $\bar{x}$ to be surrounded by the distribution of X given $Y = 0$ as defined in Section 3. Such a requirement is stronger than

N	α	Ne^{α}	β	Ne^{β}
10	-2.36	0.94100	0.1222260	1.2222
100	-4.60	0.99524	0.0097523	0.9752
1,000	-6.90	0.99953	0.0009537	0.9536
10,000	-9.21	0.99995	0.0000952	0.9515
100,000	-11.51	0.99999	0.0000095	0.9513

Table 2: Logistic regression intercept α and coefficient β for imbalanced data described in the text. There are N observations with $Y = 0$ and stratified X from the standard Cauchy distribution, and one observation with $Y = 1$ and $X = 1$.

N	α	Ne^{α}	β	e^{β}/N
10	-3.82	0.2184	2.85	1.74
100	-7.13	0.0804	4.19	0.66
1,000	-10.71	0.0223	5.82	0.34
10,000	-14.52	0.0050	7.62	0.20
100,000	-18.49	0.0009	9.54	0.14

Table 3: Logistic regression intercept α and coefficient β for imbalanced data described in the text. There are N observations with $Y=0$ and stratified $X \sim U(0,1)$ and two observations with $Y=1$, one with $X=1/2$, the other with $X=2$.

what is needed in the finite sample setting. Empirically e^{α} and e^{β} both appear to follow a power law in N but we do not investigate this further, focusing instead on the case where β approaches a non-trivial limit.

3. Notation

The data are (x,y) pairs where $x \in \mathbb{R}^d$ and $y \in \{0,1\}$. There are n observations with $y=1$ and N with $y=0$. The difference in case serves to remind us that $n \ll N$. The values of x when $y=1$ are $x_{11},\dots,x_{1n}$. The values of x when $y=0$ are $x_{01},\dots,x_{0N}$. Singly subscripted values x_i represent x_{1i}. Sometimes we use n_1 for n and n_0 for N.

The logistic regression model is $\Pr(Y=1 \mid X=x) = e^{\alpha+x'\beta}/(1+e^{\alpha+x'\beta})$ for $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}^d$. The log-likelihood in logistic regression is

$$\sum_{i=1}^{n}\Big\{\alpha + x_{1i}'\beta - \log(1+e^{\alpha+x_{1i}'\beta})\Big\} - \sum_{i=1}^{N}\log(1+e^{\alpha+x_{0i}'\beta}). \tag{3}$$

We suppose that a good approximation can be found for the conditional distribution of X given that $Y=0$, as seems reasonable when N is very large. For continuously distributed X we might then replace the second sum in (3) by $N\int \log(1+\exp(1+e^{\alpha+x'\beta})f_0(x)\,dx$ where f_0 is a probability density function. Because some or all of the components of X might be discrete we work instead with a distribution function F_0 for X given that $Y=0$.

With a bit of foresight we also center the logistic regression around the average $\bar{x} = \sum_{i=1}^{n} x_i/n$ of the predictor values for cases with $Y=1$. Then the log likelihood we work with simplifies to

$$\ell(\alpha,\beta) = n\alpha - \sum_{i=1}^{n}\log(1+e^{\alpha+(x_i-\bar{x})'\beta}) - N\int \log(1+e^{\alpha+(x-\bar{x})'\beta})\,dF_0(x) \tag{4}$$

where the $n\alpha$ term arises as $\sum_{i=1}^{n} \alpha + (x_i-\bar{x})'\beta$.

When the centered log likelihood ℓ has an MLE $(\hat{\alpha}_0,\hat{\beta})$ we can recover the MLE of the uncentered log likelihood easily: $\hat{\beta}$ remains unchanged while $\hat{\alpha}$ in the uncentered version is $\hat{\alpha}_0 - \bar{x}'\hat{\beta}$. The numerical examples in Section 2 used uncentered logistic regression.

Here we study the maximizer $(\hat{\alpha},\hat{\beta})$ of (4) in the limit as $N \to \infty$ with n and $x_1,\dots,x_n$ held fixed. It is reasonable to suppose that $\hat{\alpha} \to -\infty$ in this limit. Indeed we anticipate $e^{\hat{\alpha}}$ should be $O(1/N)$ since the proportion of observations with $y=1$ in the data is $n/(N+n) \doteq n/N$. What is interesting and important is that $\hat{\beta}$ does not necessarily diverge.

3.1 Silvapulle's Results

It is well known that the MLE in the usual logistic regression setting can fail to be finite when the x values where $y = 1$ are linearly separable from those where $y = 0$.

The existence and uniqueness of MLE's for linear logistic regression has been completely characterized by Silvapulle (1981). He works in terms of binary regression through the origin. To employ an intercept, one uses the usual device of adjoining a predictor component that is always equal to 1.

For $y = 1$ let $z_{1i} = (1, x'_{1i})'$ for $i = 1, \dots, n_1$ and for $y = 0$ let $z_{0i} = (1, x'_{0i})'$ for $i = 1, \dots, n_0$. Let $\theta = (\alpha, \beta')'$. Then the logistic regression model has $\Pr(Y = 1 \mid X = x) = \exp(z'\theta)/(1 + \exp(z'\theta))$ where of course $z = z(x) = (1, x')'$. Silvapulle (1981) employs two convex cones:

$$C_j = \left\{ \sum_{i=1}^{n_j} k_{ji} z_{ji} \mid k_{ji} > 0 \right\}, \quad j \in \{0, 1\}.$$

Theorem 1 *For data as described above, assume that the $n_0 + n_1$ by $d + 1$ matrix with rows taken from z_{ji} for $j = 0, 1$ and $i = 1, \dots, n_j$ has rank $d + 1$. If $C_0 \cap C_1 \neq \emptyset$ then a unique finite logistic regression MLE $\hat\theta = (\hat\alpha, \hat\beta')$ exists. If however $C_0 \cap C_1 = \emptyset$ then no MLE exists.*

Proof: This result follows from clause (iii) of the Theorem on page 311 of Silvapulle (1981). $\square$

Silvapulle (1981) has more general results. Theorem 1 also holds when the logistic CDF $G(t) = \exp(t)/(1 + \exp(t))$ is replaced by the standard normal one (for probit analysis) or by the $U(0, 1)$ CDF. Any CDF G for which both $-\log G(t)$ and $-\log(1 - G(t))$ are convex, and for which $G(t)$ is strictly increasing when $0 < G(t) < 1$ obeys the same theorem. The CDF G cannot be the Cauchy CDF, because the Cauchy CDF fails the convexity conditions.

The cone intersections may seem unnatural. A more readily interpretable condition is that the relative interior (as explained below) of the convex hull of the x's for $y = 0$ intersects that for $y = 1$. That is $H_0 \cap H_1 \neq \emptyset$ where

$$H_j = \left\{ \sum_{i=1}^{n_j} \lambda_{ji} x_{ji} \mid \lambda_{ji} > 0, \sum_{i=1}^{n_j} \lambda_{ji} = 1 \right\}.$$

When the x_{ji} span $\mathbb{R}^d$ then H_j is the interior of the convex hull of x_{ji}. When x_{ji} lie in a lower dimensional affine subspace of $\mathbb{R}^d$ then the interior of their convex hull is the empty set. However the interior with respect to that subspace, called the relative interior, and denoted H_j above is not empty. In the extreme where $x_{ji} = x_{j1}$ for $i = 1, \dots, n_j$, then the desired relative interior of the convex hull of $x_{j1}, \dots, x_{jn_j}$ is simply $\{x_{j1}\}$.

Lemma 2 *In the notation above $H_0 \cap H_1 \neq \emptyset$ if and only if $C_0 \cap C_1 \neq \emptyset$.*

Proof: Suppose that $x_0 \in H_0 \cap H_1$. Then $z_0 = (1, x'_0)' \in C_0 \cap C_1$. Conversely suppose that $z_0 \in C_0 \cap C_1$. Then we may write

$$z_0 = \sum_{i=1}^{n_0} k_{0i} \begin{pmatrix} 1 \\ x_{0i} \end{pmatrix} = \sum_{i=1}^{n_1} k_{1i} \begin{pmatrix} 1 \\ x_{1i} \end{pmatrix},$$

where each $k_{ji} > 0$. From the first component of z_0 we find a common positive value for $\sum_{i=1}^{n_0} k_{0i}$ and $\sum_{i=1}^{n_1} k_{1i}$. Let K denote that value, and put $\lambda_{ji} = k_{ji}/K$ for $j = 0, 1$ and $i = 1, \dots, n_j$. Then $x_0 = \sum_{i=1}^{n_0} \lambda_{0i} x_{0i} = \sum_{i=1}^{n_1} \lambda_{1i} x_{1i} \in H_0 \cap H_1$. $\square$

3.2 Overlap Conditions

In light of Silvapulle's results we expect that we will need to assume some overlap between the data $x_1,\dots,x_n$ from the 1s and the distribution F_0 of X given $Y=0$ in order to get a nontrivial result. The setting here with $N\to\infty$ is different and requires a stronger, but still very weak, overlap condition. In describing this condition, we let $\Omega=\{\omega\in\mathbb{R}^d \mid \omega'\omega=1\}$ be the unit sphere in $\mathbb{R}^d$.

Definition 3 *The distribution F on $\mathbb{R}^d$ has the point x_* surrounded if*

$$\int_{(x-x_*)'\omega>\varepsilon} dF(x)>\delta \tag{5}$$

holds for some $\varepsilon>0$, some $\delta>0$ and all $\omega\in\Omega$.

We will make use of two simple immediate consequences of (5). If F has the point x_* surrounded, then there exist η and γ satisfying

$$\inf_{\omega\in\Omega}\int_{(x-x_*)'\omega\geq 0} dF(x)\geq\eta>0 \tag{6}$$

and

$$\inf_{\omega\in\Omega}\int[(x-x_*)'\omega]_+\,dF(x)\geq\gamma>0 \tag{7}$$

where $Z_+=\max(Z,0)$ is the positive part of Z. For example we can take $\eta=\delta$ in (6) and $\gamma=\varepsilon\delta$ in (7). Notice that F cannot surround any point if F concentrates in a low dimensional affine subset of $\mathbb{R}^d$. This implies that having at least one point surrounded by F_0 will be enough to avoid rank deficiency.

In Theorem 1 it follows from Lemma 2 that we only need there to be some point x_* that is surrounded by both $\hat{F}_0$ and $\hat{F}_1$ where $\hat{F}_j$ is the empirical distribution of $x_{j1},\dots,x_{jn_j}$. If such x exists, we get a unique finite MLE. (Recall that Theorem 1 assumes full rank for the predictors.)

In the infinitely imbalanced setting we expect that F_0 will ordinarily surround every single one of $x_1,\dots,x_n$. We do not need F_0 to surround them all but it is not enough to just have some point x_* exist that is surrounded by both F_0 and $\hat{F}_1$. We need to assume that F_0 surrounds $\bar{x}$. We do not need to assume that $\hat{F}_1$ surrounds $\bar{x}$, a condition that fails when the x_i are confined to an affine subset of $\mathbb{R}^d$ as they necessarily are for $n<d$.

There is an interesting case in which F_0 can fail to surround $\bar{x}$. The predictor X may contain a component that is itself an imbalanced binary variable, and that component might never take the value 1 in the $y=1$ sample. Then $\bar{x}$ is right on the boundary of the support of F_0 and we cannot be sure of a finite β in either the finite sample case or the infinitely imbalanced case.

3.3 Technical Lemmas

The first technical lemma below is used to get some bounds. The second one establishes existence of a finite MLE when $N<\infty$.

Lemma 4 *For $\alpha,z\in\mathbb{R}$,*

$$\begin{aligned} e^{\alpha+z}&\geq\log(1+e^{\alpha+z})\geq[\log(1+e^\alpha)+ze^\alpha/(1+e^\alpha)]_+\\ &\geq[ze^\alpha/(1+e^\alpha)]_+=z_+e^\alpha/(1+e^\alpha). \end{aligned} \tag{8}$$

Proof: For the leftmost inequality, apply $x \ge \log(1+x)$ to $x = e^{\alpha+z}$. For the others, the function $h(z) = \log(1+e^{\alpha+z})$ is convex and positive. Therefore $h(z) \ge [h(0)+zh'(0)]_+ \ge [zh'(0)]_+ = z_+h'(0)$. □

Lemma 5 *Let $n \ge 1$ and $x_1,\dots,x_n \in \mathbb{R}^d$ be given, and assume that the distribution F_0 surrounds $\bar{x} = \sum_{i=1}^n x_i/n$ and that $0 < N < \infty$. Then the log likelihood $\ell(\alpha,\beta)$ given by* (4) *has a unique finite maximizer* $(\hat{\alpha},\hat{\beta})$.

Proof: The log likelihood ℓ is strictly concave in (α,β). It either has a unique finite maximizer or it grows forever along some ray $\{(\lambda\alpha_0,\lambda\beta_0) \mid 0 \le \lambda < \infty\} \subset \mathbb{R}^{d+1}$. By following such a ray back to where it intersects a small cylinder around the origin we may assume that either $0 \le |\alpha_0| < \varepsilon/2$ and $\beta_0'\beta_0 = 1$, where ε is the constant in Definition 3, or that $0 < |\alpha_0| < \varepsilon/2$ and $\beta_0 = 0$. We will show that $\partial\ell(\lambda\alpha_0,\lambda\beta_0)/\partial\lambda$ is always strictly negative, ruling out infinite growth and thus establishing a unique finite maximizer.

For $\beta_0 = 0$ and $\alpha_0 > 0$ we find $\lim_{\lambda\to\infty} \partial\ell(\lambda\alpha_0,\lambda\beta_0)/\partial\lambda = -N\alpha_0 < 0$. For $\beta_0 = 0$ and $\alpha_0 < 0$ we find $\lim_{\lambda\to\infty} \partial\ell(\lambda\alpha_0,\lambda\beta_0)/\partial\lambda = n\alpha_0 < 0$.

Now suppose $\beta_0'\beta_0 = 1$ and $|\alpha_0| < \varepsilon/2$. Using $n\alpha_0 = \sum_{i=1}^n \alpha_0 + (x_i-\bar{x})'\beta_0$, we find

$$\begin{aligned}\lim_{\lambda\to\infty} \frac{\partial}{\partial\lambda}\ell(\lambda\alpha_0,\lambda\beta_0) = &\sum_{i:\alpha_0+(x_i-\bar{x})'\beta_0<0} \alpha_0 + (x_i-\bar{x})'\beta_0 \\ &-N\int_{\alpha_0+(x-\bar{x})'\beta_0>0} \alpha_0 + (x-\bar{x})'\beta_0\, dF_0(x).\end{aligned} \tag{9}$$

The sum in (9) is either 0 or is negative and the integral is either 0 or is positive. For the integral to be 0 we must have $(x-\bar{x})'\beta_0 \le -\alpha_0$ with probability one for $x \sim F_0$. But this is impossible because F_0 has $\bar{x}$ surrounded. □

4. Main Results

Lemma 6 below shows that, as anticipated, $e^{\hat{\alpha}}$ is typically $O(1/N)$ as $N \to \infty$. Specifically, we find a bound $B = 2n/\eta < \infty$ for which $\limsup_{N\to\infty} Ne^{\hat{\alpha}} < B$.

Lemma 6 *Under the conditions of Lemma 5, let $\hat{\alpha}$ and $\hat{\beta}$ maximize ℓ of* (4). *Let η satisfy* (6). *Then for $N \ge 2n/\eta$ we have $e^{\hat{\alpha}} \le 2n/(N\eta)$.*

Proof: Let β be any point in $\mathbb{R}^d$. Write $e^{\alpha} = A/N$ for $0 < A < \infty$. Then

$$\begin{aligned}\frac{\partial}{\partial\alpha}\ell &= n - \sum_{i=1}^n \frac{AN^{-1}e^{(x_i-\bar{x})'\beta}}{1+AN^{-1}e^{(x_i-\bar{x})'\beta}} - N\int \frac{AN^{-1}e^{(x-\bar{x})'\beta}}{1+AN^{-1}e^{(x-\bar{x})'\beta}}\, dF_0(x) \\ &\le n - A\int_{(x-\bar{x})'\beta\ge 0} \frac{e^{(x-\bar{x})'\beta}}{1+AN^{-1}e^{(x-\bar{x})'\beta}}\, dF_0(x) \\ &\le n - \frac{A\eta}{1+A/N}.\end{aligned}$$

Now suppose that $N \geq 2n/\eta$ and that $e^{\alpha} > 2n/(N\eta)$, that is $A > 2n/\eta$. Then $\partial\ell/\partial\alpha < 0$. Because ℓ is concave this negative partial derivative implies that

$$\arg\max_{\alpha} \ell(\alpha,\beta) < \log(2n/\eta) - \log(N). \tag{10}$$

Because β was arbitrary (10) holds for all $\beta \in \mathbb{R}^d$. Lemma 5 implies that $\hat{\beta}$ is finite, and so (10) applies for $\beta = \hat{\beta}$. $\square$

Lemma 7 *Under the conditions of Lemma 5, let* $\hat{\alpha}$ *and* $\hat{\beta}$ *maximize* ℓ *of* (4). *Then* $\limsup_{N\to\infty} \|\hat{\beta}\| < \infty$.

Proof: Let $e^{\alpha} = A/N$ for $A > 0$ and let $\beta \in \mathbb{R}^d$. Pick γ to satisfy (7). Then

$$\begin{aligned}
&\ell(\alpha,0) - \ell(\alpha,\beta)\\
&\quad= -(n+N)\log(1+e^{\alpha}) + \sum_{i=1}^{n} \log(1+e^{\alpha+(x_i-\bar{x})'\beta})\\
&\qquad + N\int \log(1+e^{\alpha+(x-\bar{x})'\beta})\,dF_0(x)\\
&\quad> -(n+N)e^{\alpha} + N\frac{e^{\alpha}}{1+e^{\alpha}}\int_{(x-\bar{x})'\beta\geq 0}(x-\bar{x})'\beta\,dF_0(x)\\
&\quad\geq -(n+N)\frac{A}{N} + A\frac{\|\beta\|\gamma}{1+A/N}
\end{aligned}$$

after applying two inequalities from (8) and making some simplifications. If follows that $\ell(\alpha,\beta) < \ell(\alpha,0)$ whenever $\|\beta\| \geq \gamma^{-1}(1+A/N)(1+n/N)$. For large enough N we have $\|\hat{\beta}\| \leq 2/\gamma$, using Lemma 6 to control A. $\square$

As illustrated in Section 2, infinitely imbalanced logistic regression will be degenerate if F_0 has tails that are too heavy. We assume that

$$\int e^{x'\beta}(1+\|x\|)dF_0(x) < \infty, \quad \forall\beta \in \mathbb{R}^d. \tag{11}$$

Condition (11) is satisfied by distributions with bounded support and by light tailed distributions such as the multivariate normal distribution.

Theorem 8 *Let* $n \geq 1$ *and* $x_1,\dots,x_n \in \mathbb{R}^d$ *be fixed and suppose that* F_0 *satisfies the tail condition* (11) *and surrounds* $\bar{x} = \sum_{i=1}^n x_i/n$ *as described at* (5). *Then the maximizer* $(\hat{\alpha},\hat{\beta})$ *of* ℓ *given by* (4) *satisfies*

$$\lim_{N\to\infty} \frac{\int e^{x'\hat{\beta}} x\,dF_0(x)}{\int e^{x'\hat{\beta}}\,dF_0(x)} = \bar{x}.$$

Proof: Setting $\partial\ell/\partial\beta = 0$, dividing by $Ne^{\hat{\alpha}-\bar{x}'\hat{\beta}}$ and rearranging terms, gives

$$\int \frac{(x-\bar{x})e^{x'\hat{\beta}}}{1+e^{\hat{\alpha}+(x-\bar{x})'\hat{\beta}}}\,dF_0(x) = -\frac{1}{N}\sum_{i=1}^{n}\frac{e^{x_i'\hat{\beta}}(x_i-\bar{x})}{1+e^{\hat{\alpha}+(x_i-\bar{x})'\hat{\beta}}}. \tag{12}$$

Method	α	β_1	β_2	β_3	β_4	β_5	β_6
Original	-3.707	4.629	4.807	0.398	0.594	0.170	0.130
Single $y=1$	-10.116	4.623	4.984	0.397	0.595	0.193	0.182
$x_{1j}=\bar{x}$	-3.701	4.765	5.136	0.410	0.614	0.204	0.190
SE	0.041	0.696	0.851	0.040	0.130	0.299	0.413

Table 4: This table shows logistic regression coefficients for the chemical compound data set described in the text. The top row shows ordinary logistic regression coefficients. The second row shows the coefficients when the cases with $y=1$ are deleted and replaced by a single point $(\bar{x},1)$. The third row shows the coefficients when all 608 cases with $y=1$ are replaced by $(\bar{x},1)$. The fourth row shows standard errors for the ordinary logistic regression coefficients in the top row.

As $N\to\infty$ the right side of (12) vanishes because $\|\hat{\beta}\|$ is bounded as $N\to\infty$ by Lemma 7. Therefore the MLEs satisfy

$$\lim_{N\to\infty}\frac{\int x e^{x'\hat{\beta}}[1+e^{\hat{\alpha}+(x-\bar{x})'\hat{\beta}}]^{-1}\,dF_0(x)}{\int e^{x'\hat{\beta}}[1+e^{\hat{\alpha}+(x-\bar{x})'\hat{\beta}}]^{-1}\,dF_0(x)}=\bar{x}. \tag{13}$$

The denominator of (13) is at most $\int e^{x'\hat{\beta}}dF_0(x)$ and is at least

$$\int e^{x'\hat{\beta}}(1-e^{\hat{\alpha}+(x-\bar{x})'\hat{\beta}})dF_0(x)\to\int e^{x'\hat{\beta}}dF_0(x)$$

as $N\to\infty$ because $\alpha\to-\infty$ and $\int e^{2x'\hat{\beta}}dF_0(x)<\infty$ by the tail condition (11). Therefore the denominator of (13) has the same limit as $\int e^{x'\hat{\beta}}\,dF_0(x)$ as $N\to\infty$. Similarly the numerator has the same limit as $\int e^{x'\hat{\beta}}x\,dF_0(x)$. The limit for the denominator is finite and nonzero, and so the result follows. $\square$

5. Illustration

It is perhaps surprising that in the $N\to\infty$ limit, the logistic regression depends on $x_1,\dots,x_n$ only through $\bar{x}$. The precise configuration of those n points in $\mathbb{R}^d$ becomes unimportant. We could rotate them about $\bar{x}$, or replace each of them by $\bar{x}$, or even replace them by one single point at $\bar{x}$ with $Y=1$ and still get the same $\hat{\beta}$ in the $N\to\infty$ limit.

To investigate whether this effect can hold in finite data sets, we look at an example from Zhu et al. (2005). They study a data set with 29,812 chemical compounds on which 6 predictor variables were measured. Compounds were rated as active ($Y=1$) or inactive ($Y=0$) and only 608 of the compounds were active.

Table 4 shows the logistic regression coefficients for this data, as well as what happens to them when we replace the 608 data points (x,y) with $y=1$ by a single point at $(\bar{x},1)$, or by 608 points equal to $(\bar{x},1)$. In a centered logistic regression the point $(\bar{x},1)$ becomes $(\bar{x}-\bar{x},1)=(0,\dots,0,1)\in\mathbb{R}^{d+1}$.

The intercept changes a lot when we reduce the rare cases from 608 to 1 but otherwise the coefficients do not change importantly. Interestingly the single point version has a β vector closer to the original logistic regression than has the version with 608 points at $(\bar{x}, 1)$. The differences in β are quite small compared to the sampling uncertainty. We would reach the same conclusions about which predictors are most important in all three cases.

The linear predictor $\hat{\alpha} + \hat{\beta}'(x - \bar{x})$ was computed using the coefficients from each of these models (taking care to use the original x_i's not the versions set to $\bar{x}$.) The correlation between the linear predictor from logistic regression to that fit with all $x_i = \bar{x}$ is 0.999881. The correlation between the linear predictor from logistic regression to that fit with just one $(\bar{x}, 1)$ data point is still higher, at 0.999888. The two altered linear predictors have correlation 0.999998. Not surprisingly any two of these linear predictors plot as a virtual straight line. There will be no important differences in ROC curves, precision and recall curves or other performance measures among these three fits.

6. Discussion

This paper has focussed on establishing the limit of $\hat{\beta}$ as $N \to \infty$. This section presents some context and motivation. Section 6.1 shows these findings lead to greater understanding of how logistic regression works or fails and how to improve it. Section 6.2 shows how even after passing to the limit the resulting model makes some useful predictions. Section 6.3 illustrates the special case of F_0 that is Gaussian or a mixture of Gaussians. Section 6.4 describes how using infinitely imbalanced logistic regression may lead to cost savings in fraud detection settings.

6.1 Insight Into Logistic Regression

In the infinitely imbalanced limit, logistic regression only uses the $y = 1$ data points through their average feature vector $\bar{x}$. This limiting behavior is a property of logistic regression, not of any particular data set. It holds equally well in those problems for which logistic regression works badly as it does in problems where the Bayes rule is a logistic regression.

In the illustrative example we got almost the same logistic regression after replacing all the rare cases by a single point at $\bar{x}$. We would not expect this property for learning methods in general. For example classification trees such as those fit by CART (Breiman et al., 1984) will ordinarily change a lot if all of the $Y = 1$ cases are replaced by one or more points $(\bar{x}, 1)$.

Logistic regression only has d parameters apart from the intercept, so it is clear that it cannot be as flexible as some other machine learning methods. But knowing that those parameters are very strongly tied to the d components of $\bar{x}$ gives us insight into how logistic regression works on imbalanced problems. It is reasonable to expect better results from logistic regression when the x_{1i} are in a single tight cluster near $\bar{x}$ than when there are outliers, or when the x_{1i} points are in two well separated clusters in different directions from the bulk of F_0.

The insight also suggests things to do. For example when we detect outliers among the x_{1i}, shrinking them towards $\bar{x}$, or removing them should improve performance. When we detect sharp clusters among x_{1i} then we might fit one logistic regression per cluster, separating that cluster from the x_{0i}'s, and predict for new points by pooling the cluster specific results. Even an $O(n^2)$ clustering algorithm may be inexpensive in the $N \gg n$ setting.

6.2 Nontrivial Limiting Predictions

In the infinitely imbalanced limit with $N \to \infty$ we often find that $\hat{\beta}$ converges to a finite limit while $\hat{\alpha} \to -\infty$. This limit gives $\Pr(Y = 1 \mid X = x) \to 0$ for all x and so it gives trivial probabilities for prediction purposes. But we are often interested in probability ratios with nontrivial limits such as:

$$\frac{\Pr(\widetilde{Y}=1 \mid X=\widetilde{x})}{\Pr(Y=1 \mid X=x)} \to e^{(\widetilde{x}-x)'\beta}.$$

For example if we are presented with a number of cases of potential fraud to investigate and have limited resources then we can rank them by $x'\beta$ and investigate as many of the most likely ones as time or other costs allow.

Because this rank is derived from a probability ratio we can also take into account the monetary or other measured value of the cases. If the values of uncovering fraud in the two cases are v and $\widetilde{v}$, respectively, then we might prefer to investigate the former when $ve^{x'\beta} > \widetilde{v}e^{\widetilde{x}'\beta}$. If the costs of investigation are c and $\widetilde{c}$ then we might prefer the former when $ve^{x'\beta}/c > \widetilde{v}e^{\widetilde{x}'\beta}/\widetilde{c}$.

In active learning problems one must choose which data to gather. There are several kinds of active learning, as described in Tong (2001). The interventional setting is very similar to statistical experimental design. For example, Cohn et al. (1996) describe how to select training data for feedforward neural networks. In the selective setting, the investigator has a mix of labelled cases (both x and y known) and unlabelled cases (x known but y unknown), and must choose which of the unlabelled cases to get a label for. For example the label y might indicate whether a human expert says that a document with feature vector x is on a specific topic. In a rare event setting, finding the cases most likely to have $y=1$ is a reasonable proxy for finding the most informative cases, and one could then allocate a large part of the labelling budget to cases with high values of $x'\beta$.

6.3 Gaussian Mixtures F_0

When F_0 is a nonsingular Gaussian distribution then as remarked in the introduction, $\beta \to \Sigma_0^{-1}(\bar{x}-\mu_0)$. The effective sample size of an imbalanced data set is often considered to be simply the number of rare outcomes. The formula for β depends on the data only through $\bar{x}$, which as an average of n observations clearly has effective sample size of n. In the limit where $N \to \infty$ first and then $n \to \infty$ we get $\beta \to \Sigma_0^{-1}(\mu_1-\mu_0)$ where $\mu_j = E(X \mid Y=j)$. A confidence ellipsoid for μ_1 translates directly into one for β.

Gaussian mixture models are a flexible and widely used method for approximating distributions. They have the further advantage for the present problem that exponential tilts of Gaussian mixtures are also Gaussian mixtures. The result is a convenient expression to be solved for β.

Suppose that

$$F_0 = \sum_{k=1}^{K} \lambda_k N(\mu_k, \Sigma_k)$$

where $\lambda_k > 0$ and $\sum_{k=1}^{K}\lambda_k = 1$. If at least one of the Σ_k has full rank then F_0 will surround the point $\bar{x}$. Then the limiting β is defined through

$$\bar{x} = \frac{\sum_{k=1}^{K}\lambda_k(\mu_k+\Sigma_k\beta)e^{\beta'\mu_k+\beta'\Sigma_k\beta/2}}{\sum_{k=1}^{K}\lambda_k e^{\beta'\mu_k+\beta'\Sigma_k\beta/2}},$$

so that β is the solution to

$$0 = \sum_{k=1}^{K}\lambda_k(\mu_k+\Sigma_k\beta-\bar{x})e^{\beta'\mu_k+\beta'\Sigma_k\beta/2}. \tag{14}$$

Solving Equation (14) for β is cast as a convex optimization in Section 6.4.

6.4 Computational Costs

The exponential tilting solution to (1) is the value β for which $\int (x-\bar{x})e^{x'\beta}\,dF_0(x)=0$. That solution is more conveniently expressed as the root of

$$g(\beta)\equiv\int (x-\bar{x})e^{(x-\bar{x})'\beta}\,dF_0(x)=0. \tag{15}$$

Equation (15) is the gradient with respect to β of

$$f(\beta)=\int e^{(x-\bar{x})'\beta}\,dF_0(x),$$

which has Hessian

$$H(\beta)=\int (x-\bar{x})(x-\bar{x})'e^{(x-\bar{x})'\beta}\,dF_0(x).$$

The tilting problem (1) can be solved by finding the root of (15) which is in turn equivalent to the minimization of the convex function f.

When F_0 is modeled as a mixture F_0 of Gaussians the objective function, gradient, and Hessian needed for optimization have a simple form. They are,

$$\begin{aligned}
f(\beta)&=\sum_{k=1}^{K}\lambda_k e^{\beta'(\mu_k-\bar{x})+\beta'\Sigma_k\beta/2},\\
g(\beta)&=\sum_{k=1}^{K}\lambda_k(\widetilde{\mu}_k(\beta)-\bar{x})e^{\beta'(\mu_k-\bar{x})+\beta'\Sigma_k\beta/2},\quad\text{where,}\\
\widetilde{\mu}_k(\beta)&\equiv\mu_k+\Sigma_k\beta,\quad\text{and,}\\
H(\beta)&=\sum_{k=1}^{K}\lambda_k\Big(\Sigma_k+(\widetilde{\mu}_k(\beta)-\bar{x})(\widetilde{\mu}_k(\beta)-\bar{x})'\Big)e^{\beta'(\mu_k-\bar{x})+\beta'\Sigma_k\beta/2}.
\end{aligned}$$

The cost of solving (14) or (15) by an algorithm based on Newton's method takes $O(d^3)$ computation per iteration. By contrast, each step in iteratively reweighted least squares fitting of logistic regression takes $O((n+N)d^2)$ work. Even if one downsamples the data set, perhaps keeping only $N=5n$ randomly chosen examples from the $Y=0$ cases, the work of an iteration is $O(nd^2)$. The one time cost to fit a mixture of Gaussians includes costs of order Nd^2 to form covariance matrix estimates, or $O(nd^2)$ if one has downsampled. But after the first iteration there can be substantial computational savings for solving (15) instead of doing logistic regression, when n/d is large.

When there is one common class and there are numerous rare classes, such as types of fraud or different targets against which a drug might be active, then the cost of approximating F_0 can be shared over the set of uncommon classes.

In fraud detection problems we might expect that the distribution F_0 for legitimate data points is slowly changing while the patterns in the fraudulent points change rapidly in response to improved detection. In such a setting we get a computational saving by fitting an approximation to F_0 once, or at long time intervals, and then computing many different $\beta(\infty)$ vectors. These vectors can be for different known types of fraud, for fraud over shorter time intervals, or even individual fraud cases.

Acknowledgments

This work was supported by NSF grants DMS-0306612 and DMS-0604939. I thank Alan Agresti, Trevor Hastie for their comments. Thanks also to the JMLR reviewers for their speedy and helpful reviews. I am grateful for many insightful comments from Paul Louisell.

References

R. J. Bolton and D. J. Hand. Statistical fraud detection: A review. *Statistical Science*, 17(3):235–255, 2002.

L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification And Regression Trees*. Wadsworth, Belmont, CA, 1984.

N.V. Chawla, N. Japkowicz, and A. Kolcz. *Proceedings of the ICML'2003 Workshop on Learning from Imbalanced Data Sets*. 2003.

N.V. Chawla, N. Japkowicz, and A. Kolcz. Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, 6(1):1–6, 2004.

D.A. Cohn, Z. Ghahramani, and M.I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

N. Japkowicz. *Learning from Imbalanced Data Sets: Papers from the AAAI Workshop*. AAAI, 2000. Technical Report WS-00-05.

G. King and L. Zeng. Logistic regression in rare events data. *Political Analysis*, 9(2):137–163, 2001.

M.J. Silvapulle. On the existence of maximum likelihood estimates for the binomial response models. *Journal of the Royal Statistical Society, Series B*, 43:310–313, 1981.

S. Tong. *Active learning: Theory and applications*. PhD thesis, Stanford University, 2001. URL `http://ai.stanford.edu/∼stong/research.html/tong_thesis.pdf`.

M. Zhu, W. Su, and H. A. Chipman. LAGO: A computationally efficient approach for statistical detection. *Technometrics*, 48:193–205, 2005.

Journal of Machine Learning Research 8 (2007) 775-790 Submitted 6/06; Published 4/07

Sparseness vs Estimating Conditional Probabilities: Some Asymptotic Results

Peter L. Bartlett BARTLETT@CS.BERKELEY.EDU
Division of Computer Science and Department of Statistics
University of California
Berkeley, CA 94720-1776, USA

Ambuj Tewari AMBUJ@CS.BERKELEY.EDU
Division of Computer Science
University of California
Berkeley, CA 94720-1776, USA

Editor: Gábor Lugosi

Abstract

One of the nice properties of kernel classifiers such as SVMs is that they often produce sparse solutions. However, the decision functions of these classifiers cannot always be used to estimate the conditional probability of the class label. We investigate the relationship between these two properties and show that these are intimately related: sparseness does not occur when the conditional probabilities can be unambiguously estimated. We consider a family of convex loss functions and derive sharp asymptotic results for the fraction of data that becomes support vectors. This enables us to characterize the exact trade-off between sparseness and the ability to estimate conditional probabilities for these loss functions.

Keywords: kernel methods, support vector machines, sparseness, estimating conditional probabilities

1. Introduction

Consider the following familiar setting of a binary classification problem. A sequence $T = ((x_1,y_1), \ldots, (x_n,y_n))$ of i.i.d. pairs is drawn from a probability distribution over $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y}$ is the set of labels (which we assume is $\{+1,-1\}$ for convenience). The goal is to use the training set T to predict the label of a new observation $x \in \mathcal{X}$. A common way to approach the problem is to use the training set to construct a decision function $f_T : \mathcal{X} \to \mathbb{R}$ and output $\mathrm{sign}(f_T(x))$ as the predicted label of x.

In this paper, we consider classifiers based on an optimization problem of the form:

$$f_{T,\lambda} = \arg\min_{f \in H} \lambda \|f\|_H^2 + \frac{1}{n}\sum_{i=1}^{n} \phi(y_i f(x_i)). \tag{1}$$

Here, H is a reproducing kernel Hilbert space (RKHS) of some kernel k, $\lambda > 0$ is a regularization parameter and $\phi : \mathbb{R} \to [0,\infty)$ is a convex loss function. Since optimization problems based on the non-convex function 0-1 loss $t \mapsto I_{(t \le 0)}$ (where $I_{(\cdot)}$ is the indicator function) are computationally intractable, use of convex loss functions is often seen as using upper bounds on the 0-1 loss to make the problem computationally easier. Although computational tractability is one of the goals we have in mind while designing classifiers, it is not the only one. We would like to compare different convex loss functions based on their statistical and other useful properties. Conditions ensuring Bayes-risk consistency of classifiers using convex loss functions have already been established (Bartlett et al., 2004; Lugosi and Vayatis, 2004; Steinwart, 2005; Zhang, 2004). It has been observed that different cost functions have different properties and it is important to choose a loss function judiciously (e.g., see Wahba, 2002). In order to understand the relative merits of different loss functions, it is important to consider these properties and investigate the extent to which different loss functions exhibit them. It may turn out (as it does below) that different properties are in conflict with each other. In that case, knowing the trade-off allows one to make an informed choice while choosing a loss function for the classification task at hand.

One of the properties we focus on is the ability to estimate the conditional probability of the class label $\eta(x) = P(Y = +1|X = x)$. Under some conditions on the loss function and the sequence of regularization parameters λ_n, the solutions of (1) converge (in probability) to a function $F^*_\phi(\eta(x))$ which is set valued in general (Steinwart, 2003). As long as we can uniquely identify $\eta(x)$ based on a value in $F^*_\phi(\eta(x))$, we can hope to estimate conditional probabilities using $f_{T,\lambda_n}(x)$, at least asymptotically. Choice of the loss function is crucial to this property. For example, the L2-SVM (which uses the loss function $t \mapsto (\max\{0, 1-t\})^2$) is much better than L1-SVM (which uses $t \mapsto \max\{0, 1-t\}$) in terms of asymptotically estimating conditional probabilities.

Another criterion is the sparseness of solutions of (1). It is well known that any solution $f_{T,\lambda}$ of (1) can be represented as

$$f_{T,\lambda}(x) = \sum_{i=1}^{n} \alpha_i^* k(x, x_i) \; . \tag{2}$$

The observations x_i for which the coefficients α_i^* are non-zero are called support vectors. The rest of the observations have no effect on the value of the decision function. Having fewer support vectors leads to faster evaluation of the decision function. Bounds on the number of support vectors are therefore useful to know. Steinwart's recent work (Steinwart, 2004) has shown that for the L1-SVM and a suitable kernel, the asymptotic fraction of support vectors is twice the Bayes-risk. Thus, L1-SVMs can be expected to produce sparse solutions. It was also shown that L2-SVMs will typically not produce sparse solutions.

We are interested in how sparseness relates to the ability to estimate conditional probabilities. What we mentioned about L1 and L2-SVMs leads to several questions. Do we always lose sparseness by being able to estimate conditional probabilities? Is it possible to characterize the exact trade-off between the asymptotic fraction of support vectors and the ability to estimate conditional probabilities? If sparseness is indeed lost when we are able to fully estimate conditional probabilities, we may want to estimate conditional probabilities only in an interval, say $(0.05, 0.95)$, if that helps recover sparseness. Estimating η for x's that have $\eta(x) \geq 0.95$ may not be too crucial for our prediction task. How can we design loss functions which enable us to estimate probabilities in sub-intervals of $[0, 1]$ while preserving as much sparseness as possible?

This paper attempts to answer these questions. We show that if one wants to estimate conditional probabilities in an interval $(1-\gamma_0, \gamma_0)$ for some $\gamma_0 \in (1/2, 1)$, then sparseness is lost on that interval in the sense that the asymptotic fraction of data that become support vectors is lower bounded by $\mathbb{E}_x G(\eta(x))$ where $G(\eta) = 1$ throughout the interval $(1-\gamma_0, \gamma_0)$. Moreover, one cannot recover sparseness by giving up the ability to estimate conditional probabilities in some sub-interval of $(1-\gamma_0, \gamma_0)$. The only way to do that is to decrease γ_0 thereby shortening the interval $(1-\gamma_0, \gamma_0)$. We also derive sharp bounds on the asymptotic number of support vectors for a family of loss functions

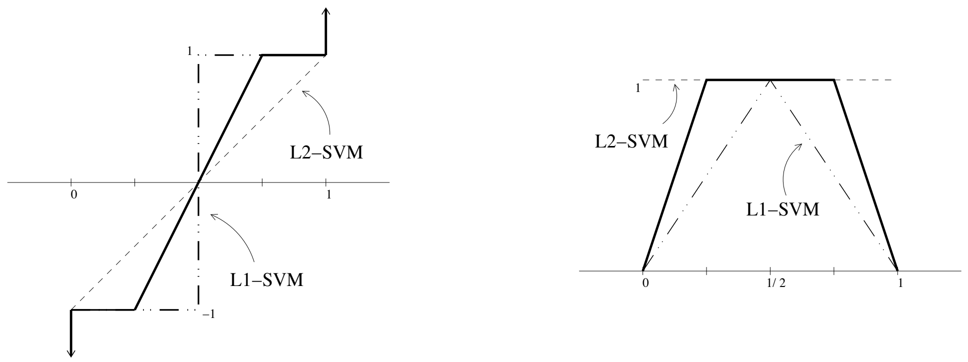


Figure 1: Plots of F^*_ϕ (left) and G (right) for a loss function which is a convex combination of the L1 and L2-SVM loss functions. Dashed lines represent the corresponding plots for the original loss functions.

of the form:

$$\phi(t) = h((t_0 - t)_+),\ t_0 > 0$$

where t_+ denotes $\max\{0,t\}$ and h is a continuously differentiable convex function such that $h'(0) \geq 0$. Each loss function in the family allows one to estimate probabilities in the interval $(1-\gamma_0, \gamma_0)$ for some value of γ_0. The asymptotic fraction of support vectors is then $\mathbb{E}_x G(\eta(x))$, where $\eta \mapsto G(\eta)$ is a function that increases linearly from 0 to 1 as η goes from 0 to $1-\gamma_0$. For example, if $\phi(t) = \frac{1}{3}((1-t)_+)^2 + \frac{2}{3}(1-t)_+$ then conditional probabilities can be estimated in $(1/4, 3/4)$ and $G(\eta) = 1$ for $\eta \in (1/4, 3/4)$ (see Fig. 1).

2. Notation and Known Results

Let P be the probability distribution over $\mathcal{X} \times \mathcal{Y}$ and let $T \in (\mathcal{X} \times \mathcal{Y})^n$ be a training set. Let $\mathbb{E}_P(\cdot)$ denote expectations taken with respect to the distribution P. Similarly, let $\mathbb{E}_x(\cdot)$ denote expectations taken with respect to the marginal distribution on $\mathcal{X}$. Let $\eta(x)$ be $P(Y=+1|X=x)$. For a decision function $f : \mathcal{X} \to \mathbb{R}$, define its risk as

$$R_P(f) = \mathbb{E}_P I_{(yf(x) \leq 0)} \ .$$

The Bayes-risk $R_P = \inf\{R_P(f) : f \text{ measurable}\}$ is the least possible risk. Given a loss function ϕ, define the ϕ-risk of f by

$$R_{\phi,P}(f) = \mathbb{E}_P \phi(yf(x)) \ .$$

The optimal ϕ-risk $R_{\phi,P} = \inf\{R_{\phi,P}(f) : f \text{ measurable}\}$ is the least achievable ϕ-risk. When the expectations in the definitions of $R_P(f)$ and $R_{\phi,P}(f)$ are taken with respect to the empirical measure corresponding to T, we get the empirical risk $R_T(f)$ and the empirical ϕ-risk $R_{\phi,T}(f)$ respectively. Conditioning on x, we can write the ϕ-risk as

$$\begin{aligned} R_{\phi,P}(f) &= E_x[E(\phi(yf(x)|x)] \\ &= E_x[\eta(x)\phi(f(x)) + (1-\eta(x))\phi(-f(x))] \\ &= E_x[C(\eta(x), f(x))] \ . \end{aligned}$$

Here, we have defined $C(\eta,t) = \eta\phi(t) + (1-\eta)\phi(-t)$. To minimize the ϕ-risk, we have to minimize $C(\eta,\cdot)$ for each $\eta \in [0,1]$. So, define the set valued function F^*_ϕ by

$$F^*_\phi(\eta) = \{t : C(\eta,t) = \min_{s\in\bar{\mathbb{R}}} C(\eta,s)\}$$

where $\bar{\mathbb{R}}$ is the set of extended reals $\mathbb{R} \cup \{-\infty,\infty\}$. Any measurable selection f^* of F^*_ϕ actually minimizes the ϕ-risk. The function F^*_ϕ is plotted for three choices of ϕ in Fig. 1. From the definitions of $C(\eta,t)$ and $F^*_\phi(\eta)$, it is easy to see that $F^*_\phi(\eta) = -F^*_\phi(1-\eta)$. Steinwart (2003) also proves that $\eta \mapsto F^*_\phi(\eta)$ is a monotone operator. This means that if $\eta_1 > \eta_2$, $t_1 \in F^*_\phi(\eta_1)$ and $t_2 \in F^*_\phi(\eta_2)$ then $t_1 \geq t_2$.

A convex loss function is called classification calibrated if the following two conditions hold:

$$\eta < \frac{1}{2} \Rightarrow F^*_\phi(\eta) \subset [-\infty,0) \text{ and } \eta > \frac{1}{2} \Rightarrow F^*_\phi(\eta) \subset (0,+\infty]\ .$$

A necessary and sufficient condition for a convex ϕ to be classification calibrated is that $\phi'(0)$ exists and is negative (Bartlett et al., 2004). If ϕ is classification calibrated then it is guaranteed that for any sequence f_n such that $R_{\phi,P}(f_n) \to R_{\phi,P}$, we have $R_P(f_n) \to R_P$. Thus, classification calibrated loss functions are good in the sense that minimizing the ϕ-risk leads to classifiers that have risks approaching the Bayes-risk. Note, however, that in the optimization problem (1), we are minimizing the regularized ϕ-risk

$$R^{reg}_{\phi,T,\lambda} = \lambda\|f\|^2_H + R_{\phi,T}\ .$$

Steinwart (2005) has shown that if one uses a classification calibrated convex loss function, a universal kernel (one whose RKHS is dense in the space of continuous functions over $\mathcal{X}$) and a sequence of regularization parameters such that $\lambda_n \to 0$ sufficiently slowly, then $R_{\phi,P}(f_{T,\lambda_n}) \to R_{\phi,P}$. In another paper (Steinwart, 2003), he proves that this is sufficient to ensure the convergence in probability of f_{T,λ_n} to $F^*_\phi(\eta(\cdot))$. That is, for all $\varepsilon > 0$

$$P_x(\{x \in \mathcal{X} : \rho(f_{T,\lambda_n}(x), F^*_\phi(\eta(x))) \geq \varepsilon\}) \to 0. \tag{3}$$

The function $\rho(t,B)$ is just the distance from t to the point in B which is closest to t. The definition given by Steinwart (2003) is more complicated because one has to handle the case when $B \cap \mathbb{R} = \emptyset$. We will ensure in our proofs that F^*_ϕ is not a singleton set just containing $+\infty$ or $-\infty$.

Since f_{T,λ_n} converges to $F^*_\phi(\eta(\cdot))$, the plots in Fig. 1 suggest that the L2-SVM decision function can be used to estimate conditional probabilities in the whole range $[0,1]$ while it not possible to use the L1-SVM decision function to estimate conditional probabilities in any interval. However, the L1-SVM is better if one considers the asymptotic fraction of support vectors. Under some conditions on the kernel and the regularization sequence, Steinwart proved that the fraction is $\mathbb{E}_x[2\min(\eta(x), 1-\eta(x))]$, which also happens to be the optimal ϕ-risk for the hinge loss function. For L2-SVM, he showed that the asymptotic fraction is $P_x(\{x \in \mathcal{X} : 0 < \eta(x) < 1\})$, which is the probability of the set where noise occurs. Observe that we can write the fraction of support vectors as $\mathbb{E}_x[G(\eta(x))]$ where $G(\eta) = 2\min\{\eta, 1-\eta\}$ for the hinge loss and $G(\eta) = I_{(\eta\notin\{0,1\})}$ for the squared hinge loss. We will see below that these two are extreme cases. In general, there are loss functions which allow one to estimate probabilities in an interval centered at 1/2 and for which $G(\eta) = 1$ only on that interval.

Steinwart (2003) also derived a general lower bound on the asymptotic number of support vectors in terms of the probability of the set

$$S = \{(x,y) \in \mathcal{X}_{cont} \times \mathcal{Y} : 0 \notin \partial\phi(yF^*_\phi(\eta(x)))\} .$$

Here, $\mathcal{X}_{cont} = \{x \in \mathcal{X} : P_x(\{x\}) = 0\}$ and $\partial\phi$ denotes the subdifferential of ϕ. In the simple case of a function of one variable $\partial\phi(x) = [\phi'_-(x), \phi'_+(x)]$, where ϕ'_- and ϕ'_+ are the left and right hand derivatives of ϕ (which always exist for convex functions). If $\mathcal{X}_{cont} = \mathcal{X}$, one can write $P(S)$ as

$$\begin{aligned} P(S) &= \mathbb{E}_P[I_{(0\notin\partial\phi(yF^*_\phi(\eta(x))))}] \\ &= \mathbb{E}_x[\eta(x) I_{(0\notin\partial\phi(F^*_\phi(\eta(x))))} + (1-\eta(x)) I_{(0\notin\partial\phi(-F^*_\phi(\eta(x))))}] \\ &= \mathbb{E}_x G(\eta(x)) . \end{aligned} \tag{4}$$

For the last step, we simply defined

$$G(\eta) = \eta I_{(0\notin\partial\phi(F^*_\phi(\eta)))} + (1-\eta) I_{(0\notin\partial\phi(-F^*_\phi(\eta)))} . \tag{5}$$

3. Preliminary Results

We will consider only classification calibrated convex loss functions. Since ϕ is classification calibrated we know that $\phi'(0) < 0$. Define t_0 as

$$t_0 = \inf\{t : 0 \in \partial\phi(t)\}$$

with the convention that $\inf\emptyset = \infty$. Because $\phi'(0) < 0$ and subdifferentials of a convex function are monotonically decreasing, we must have $t_0 > 0$. However, it may be that $t_0 = \infty$. The following lemma says that sparse solutions cannot be expected if that is the case.

Lemma 1 *If* $t_0 = \infty$*, then* $G(\eta) = 1$ *for* $\eta \in [0,1]$.

Proof $t_0 = \infty$ implies that for all t, $0 \notin \partial\phi(t)$. Using (5), we get $G(\eta) = \eta.1 + (1-\eta).1 = 1$. ■

Thus, for losses like the exponential loss $t \mapsto \exp(-t)$, the bound (4) says that the fraction of support vectors approaches 1 asymptotically. Since we are interested in loss functions that lead to sparse solutions, let us assume that

$$\text{(A1)} \qquad t_0 < \infty .$$

Although not immediate from the definition, a continuity argument involving the one-sided derivatives gives us $0 \in \partial\phi(t_0)$.

We now proceed to investigate the general form of the function $\eta \mapsto F^*_\phi(\eta)$. Before we begin, we make another assumption about the number of points in the interval $(-t_0, t_0)$ where the function ϕ fails to be differentiable. Define the set

$$D = \{t \in (0, t_0) : \text{ either } \phi'(t) \text{ or } \phi'(-t) \text{ does not exist } \} .$$

We assume that

$$\text{(A2)} \qquad |D| < \infty .$$

Note that, since ϕ is convex, we know that D can be at most countably infinite. Let $t_1 > t_2 > \ldots > t_L$ be the set D sorted in decreasing order. We have $t_0 > t_1$ and $t_L > 0$. Set $\beta_0 = 1$ and define

$$\beta_l = \frac{1}{1 + \frac{\phi'_+(t_l)}{\phi'_-(-t_l)}} \ , \ \ 1 \le l \le L \ ,$$

$$\gamma_l = \frac{1}{1 + \frac{\phi'_-(t_l)}{\phi'_+(-t_l)}} \ , \ \ 0 \le l \le L \ .$$

With the possible exception of $\phi'_-(t_0)$ (which can be zero), all one-sided derivatives appearing in the definitions above are negative. For $s < t$, we have $\phi'_-(s) \le \phi'_+(s) \le \phi'_-(t) \le \phi'_+(t)$. Using this fact, we get

$$\frac{1}{2} \le \gamma_L \le \beta_L \le \ldots \le \gamma_1 \le \beta_1 \le \gamma_0 \le \beta_0 = 1 \ .$$

Since $t_l \in D$ for $1 \le l \le L$, at least one of the inequalities, $\phi'_-(t_l) < \phi'_+(t_l)$ or $\phi'_-(-t_l) < \phi'_+(-t_l)$, is strict. So $\gamma_l < \beta_l$ for $1 \le l \le L$ and we get

$$\frac{1}{2} \le \gamma_L < \beta_L \le \ldots \le \gamma_1 < \beta_1 \le \gamma_0 \le \beta_0 = 1 \ .$$

Using the observation that $F^*_\phi(1-\eta) = -F^*_\phi(\eta)$, we restrict ourselves to examining the behavior of $F^*_\phi(\eta)$ on an interval containing $[1/2, 1]$.

Theorem 2 *Let ϕ be a classification calibrated convex loss function satisfying assumptions (A1) and (A2). Then the following statements hold true.*

1. *For $0 \le l \le L$ and for $\eta \in (\gamma_l, \beta_l)$, $F^*_\phi(\eta) = \{t_l\}$.*
2. *If I is one of the (possibly degenerate) intervals $[\beta_l, \gamma_{l-1}]$, $1 \le l \le L$, then there exists a continuous non-decreasing function g_I mapping $[t_l, t_{l-1}]$ onto I such that, for $\eta \in I$,*

$$F^*_\phi(\eta) = g_I^{-1}(\eta) \ ,$$

   where $g_I^{-1}(\eta) = \{t : g_I(t) = \eta\}$.
3. *The above also holds for the (possibly degenerate) interval $I = [1-\gamma_L, \gamma_L]$ but here the function g_I maps $[-t_L, t_L]$ onto I.*
4. *$F^*_\phi(1)$ is either $[t_0, t']$, for some $t' \ge t_0$, or $[t_0, \infty)$.*

Proof (Part 1) Denoting the subdifferential with respect to the second argument by ∂_2, we have

$$\begin{aligned} \partial_2 C(\eta, t_l) &= \eta \partial\phi(t_l) - (1-\eta)\partial\phi(-t_l) \\ &= [\eta\phi'_-(t_l) - (1-\eta)\phi'_+(-t_l), \eta\phi'_+(t_l) - (1-\eta)\phi'_-(-t_l)] \ . \end{aligned}$$

For $\gamma_l < \eta < \beta_l$, we have

$$\eta\phi'_-(t_l) - (1-\eta)\phi'_+(-t_l) < 0 < \eta\phi'_+(t_l) - (1-\eta)\phi'_-(-t_l) \ ,$$

and so t_l is the unique minimizer of $C(\eta,\cdot)$.

(Part 2) Let $I = [\beta_l, \gamma_{l-1}]$. Note that for $t \in (t_l, t_{l-1})$, both $\phi'(t)$ and $\phi'(-t)$ are defined. Being derivatives of a convex function, they are continuous. So we define, for $t \in (t_l, t_{l-1})$,

$$g_I(t) = \frac{1}{1+\frac{\phi'(t)}{\phi'(-t)}} \ .$$

Convexity also implies that $g_I(t)$ is monotonically increasing. In fact, it will be strictly increasing if one of $\phi(t)$ and $\phi(-t)$ is strictly convex. However, if both $\phi(t)$ and $\phi(-t)$ are linear on some open interval, $g_I(t)$ will be constant over that interval. Define $g_I(t_l) = \beta_l$ and $g_I(t_{l-1}) = \gamma_{l-1}$. Since $\lim_{t\downarrow t_l} \phi'(t) = \phi'_+(t_l)$ and $\lim_{t\downarrow t_l} \phi'(-t) = \phi'_-(-t_l)$ (e.g., see Rockafellar, 1970, Chap. 24), we have $\lim_{t\downarrow t_l} g_I(t) = \beta_l = g_I(t_l)$. Similarly, $\lim_{t\uparrow t_{l-1}} g_I(t) = \gamma_{l-1} = g_I(t_{l-1})$. So, we have defined a continuous monotonically increasing function on $[\beta_l, \gamma_{l-1}]$ onto $[t_l, t_{l-1}]$. If $t \in (t_l, t_{l-1})$ then, for any η,

$$\begin{aligned} t \in F^*_\phi(\eta) &\text{ iff } \eta\phi'(t) + (1-\eta)\phi'(-t) = 0 \\ &\text{iff } \frac{1}{1+\frac{\phi'(t)}{\phi'(-t)}} = \eta \\ &\text{iff } g_I(t) = \eta \\ &\text{iff } t \in g_I^{-1}(\eta) \ . \end{aligned}$$

It remains to handle t_l and t_{l-1}. For $\eta \in I$, $t_l \in F^*_\phi(\eta)$ iff $\eta = \beta_l$. Since $g_I(t_l) = \beta_l$, we also have $t_l \in g_I^{-1}(\eta)$ iff $\eta = \beta_l$. Arguing similarly for t_{l-1}, for $\eta \in I$, $t_l \in F^*_\phi(\eta)$ iff $\eta = \gamma_{l-1}$. Since $g_I(t_{l-1}) = \gamma_{l-1}$, we also have $t_{l-1} \in g_I^{-1}(\eta)$ iff $\eta = \gamma_{l-1}$.

(Part 3) Once we observe that both $\phi'(t)$ and $\phi'(-t)$ exist for $t \in (-t_L, t_L)$, the proof proceeds as in part 2.

(Part 4) Let $t' = \sup\{t : 0 \in \partial\phi(t)\}$. If the set in question is unbounded, then $0 \in \partial\phi(t)$ for all $t \geq t_0$ and so $F^*_\phi(1) = [t, \infty)$. If the set is bounded above then the supremum t' is well defined. The t's which minimize $C(1,\cdot) = \phi(t)$ are then precisely those in the interval $[t_0, t']$. ■

The above theorem says a couple of interesting things about the relationship between ϕ and F^*_ϕ. Points of non-differentiability of ϕ lead to the function F^*_ϕ being constant on certain intervals. On an interval I where F^*_ϕ is not constant, there exists a function g_I such that given $t \in F^*_\phi(\eta)$, one can recover η by the relation $\eta = g_I(t)$. We can express this by saying that $F^*_\phi(\eta)$ is invertible on intervals that correspond to differentiable portions of $\phi(t)$ and $\phi(-t)$ (see Fig. 2 for an example).

Theorem 3 *Let ϕ be an classification calibrated convex loss function satisfying assumptions (A1) and (A2). Then, for $G(\eta)$ as defined in* (5)*, we have*

$$G(\eta) = \begin{cases} 1 & \eta \in (1-\gamma_0, \gamma_0) \ , \\ \min\{\eta, 1-\eta\} & \eta \in [0, 1-\gamma_0] \cup [\gamma_0, 1] \ , \end{cases} \tag{6}$$

where $\gamma_0 = \phi'_+(-t_0)/(\phi'_+(-t_0) + \phi'_-(t_0))$.

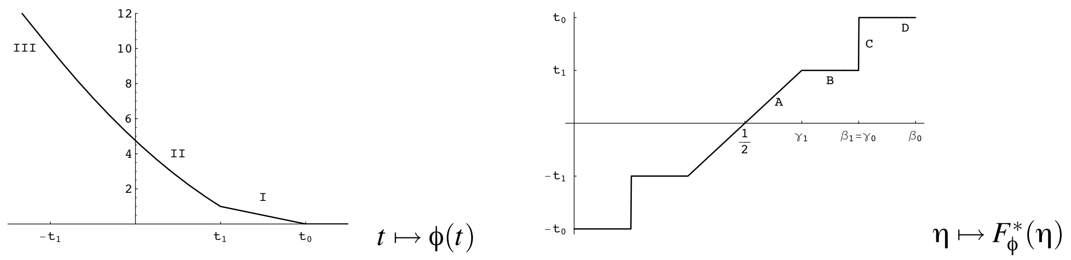


Figure 2: The loss function (left) is composed of two linear parts (I & III) and a quadratic part (II). The function F_ϕ^* (right) is constant on the regions marked B and D correponding to the 2 points of non-differentiability, viz. t_0, t_1. The vertical part C is due to ϕ being linear on the intervals $[-t_0, -t_1]$ and $[t_1, t_0]$. Region A arises due to the quadratic part of the loss function.

Proof If $\eta \in [\gamma_0, 1]$, Theorem 2 tells us that $t_0 \in F_\phi^*(\eta)$ and hence $0 \in \partial\phi(F_\phi^*(\eta))$. If $\eta < 1-\gamma_0$, monotonicity of $F_\phi^*(\eta)$ implies that $F_\phi^*(\eta) \subseteq (-\infty, t_0)$. Since $t_0 = \inf\{t : 0 \in \partial\phi(t)\}, 0 \notin \partial\phi(F_\phi^*(\eta))$ for $\eta \in [0, \gamma_0)$. Thus, we can write $I_{(0\notin\partial\phi(F_\phi^*(\eta)))}$ as $I_{(\eta\notin[\gamma_0,1])}$. Also $I_{(0\notin\partial\phi(-F_\phi^*(\eta))} = I_{(0\notin\partial\phi(F_\phi^*(1-\eta))}$. Plugging this in (5), we get

$$\begin{aligned} G(\eta) &= \eta I_{(\eta\notin[\gamma_0,1])} + (1-\eta) I_{(1-\eta\notin[\gamma_0,1])} \\ &= \eta I_{(\eta\notin[\gamma_0,1])} + (1-\eta) I_{(\eta\notin[0,1-\gamma_0])} \ . \end{aligned}$$

Since $\gamma_0 \geq 1/2$, we can write $G(\eta)$ in the form given above. ■

Corollary 4 *If* $\eta_1 \in (0,1)$ *is such that* $F_\phi^*(\eta_1) \cap F_\phi^*(\eta) = \emptyset$ *for* $\eta \neq \eta_1$*, then* $G(\eta) = 1$ *on* $[\min\{\eta_1, 1-\eta_1\}, \max\{\eta_1, 1-\eta_1\}]$.

Proof Theorem 1, part 1 tells us that such an η_1 cannot lie in $[0, 1-\gamma_0] \cup [\gamma_0, 1]$. Rest follows from Theorem 3. ■

The preceding theorem and corollary have important implications. First, we can hope to have sparseness only for values of $\eta \in [0, 1-\gamma_0] \cup [\gamma_0, 1]$. Second, we cannot estimate conditional probabilities in these two intervals because $F_\phi^*(\cdot)$ is not invertible there. Third, any loss function for which $F_\phi^*(\cdot)$ is invertible, say at $\eta_1 < 1/2$, will necessarily not have sparseness on the interval $[\eta_1, 1-\eta_1]$.

Note that for the case of L1 and L2-SVM, γ_0 is $1/2$ and 1 respectively. For these two classifiers, the lower bounds $\mathbb{E}_x G(\eta(x))$ obtained after plugging in γ_0 in (6) are the ones proved initially (Steinwart, 2003). For the L1-SVM, the bound was later significantly improved (Steinwart, 2004). This suggests that $\mathbb{E}_x G(\eta(x))$ might be a loose lower bound in general. In the next section we will show, by deriving a sharp asymptotic result, that the bound is indeed loose for a family of loss functions.

4. Asymptotic Fraction of Support Vectors

We will consider convex loss functions of the form

$$\phi(t) = h((t_0 - t)_+) \ . \tag{7}$$

The function h is assumed to be continuously differentiable and convex. We also assume $h'(0) > 0$. The convexity of ϕ requires that $h'(0)$ be non-negative. The reason we rule out $h'(0) = 0$ is that $\gamma_0 = 1$ in that case and Theorem 3 already tells us we will not have any sparseness. So, in this section we are not interested in loss functions that are differentiable everywhere. In other words, our assumption is that the loss function is constant for all $t \geq t_0$ and is continuously differentiable to the left of t_0. Further, the only discontinuity in the derivative is at t_0. Without loss of generality, we may assume that $h(0) = 0$ because the solutions to (1) do not change if we add or subtract a constant from ϕ. Note that we obtain the hinge loss if we set $h(t) = t$. We now derive the dual of (1) for our choice of the loss function.

4.1 Dual Formulation

For a convex loss function $\phi(t) = h((t_0 - t)_+)$, consider the optimization problem:

$$\arg\min_w \lambda \|w\|^2 + \frac{1}{n}\sum_{i=1}^n \phi(y_i w^T x_i) \ .$$

Make the substitution $\xi_i = t_0 - y_i w^T x_i$ to get

$$\begin{gathered} \arg\min_w \lambda \|w\|^2 + \frac{1}{n}\sum_{i=1}^n \phi(t_0 - \xi_i) \\ \text{subject to } \xi_i = t_0 - y_i w^T x_i \text{ for all } i \ . \end{gathered} \tag{8}$$

Introducing Lagrange multipliers, we get the Lagrangian:

$$\mathcal{L}(w, \xi, \alpha) = \lambda \|w\|^2 + \frac{1}{n}\sum_{i=1}^n \phi(t_0 - \xi_i) + \sum_{i=1}^n \alpha_i (t_0 - y_i w^T x_i - \xi_i) \ .$$

Minimizing this with respect to the primal variables w and ξ_i's, gives us

$$\begin{aligned} w &= \frac{1}{2\lambda}\sum_{i=1}^n \alpha_i y_i x_i \ , \\ \alpha_i &\in -\partial\phi(t_0 - \xi_i)/n \ . \end{aligned}$$

For the specific form of ϕ that we are working with, we have

$$-\partial\phi(t_0 - \xi_i)/n = \begin{cases} \{h'(\xi_i)/n\} & \xi_i > 0 \ , \\ [0, h'(0)/n] & \xi_i = 0 \ , \\ \{0\} & \xi_i < 0 \ . \end{cases} \tag{9}$$

Let (w^*, ξ_i^*) be a solution of (8). Then we have

$$\begin{aligned} \lambda \|w^*\|^2 &= \lambda (w^*)^T \left(\frac{1}{2\lambda}\sum_{i=1}^n \alpha_i^* y_i x_i \right) \\ &= \frac{1}{2}\sum_{i=1}^n \alpha_i^* y_i (w^*)^T x_i = \frac{1}{2}\sum_{i=1}^n \alpha_i^* (t_0 - \xi_i^*) \ . \end{aligned} \tag{10}$$

4.2 Result About Asymptotic Fraction of Support Vectors and Its Proof

Recall that a kernel is called universal if its RKHS is dense in the space of continuous functions over $\mathcal{X}$. Suppose the kernel k is universal and analytic. This ensures that any function in the RKHS H of k is analytic. Following Steinwart (2004), we call a probability distribution P non-trivial (with respect to ϕ) if

$$R_{\phi,P} < \inf_{b \in \mathbb{R}} R_{\phi,P}(b) \ .$$

We also define the P-version of the optimization problem (1):

$$f_{P,\lambda} = \arg\min_{f \in H} \lambda \|f\|_H^2 + E_P \phi(yf(x)) \ .$$

Further, suppose that $K = \sup\{\sqrt{k(x,x)} : x \in \mathcal{X}\}$ is finite. Fix a loss function of the form (7). Let G be the function defined as

$$G(\eta) = \begin{cases} \eta/(1-\gamma_0) & 0 \le \eta \le 1-\gamma_0 \ , \\ 1 & 1-\gamma_0 < \eta < \gamma_0 \ , \\ (1-\eta)/(1-\gamma_0) & \gamma_0 \le \eta \le 1 \ , \end{cases}$$

where $\gamma_0 = h'(2t_0)/(h'(0)+h'(2t_0))$. Note that this definition is different from the one given in (5).

Since ϕ is differentiable on $(-t_0, t_0)$, Theorem 2, part 2 implies that F_ϕ^* is invertible on $(1-\gamma_0, \gamma_0)$. Thus, one can estimate conditional probabilities in the interval $(1-\gamma_0, \gamma_0)$. Let $\#SV(f_{T,\lambda})$ denote the number of support vectors in the solution (2):

$$\#SV(f_{T,\lambda}) = |\{i : \alpha_i^* \neq 0\}| \ .$$

The next theorem says that the fraction of support vectors converges to the expectation $\mathbb{E}_x G(\eta(x))$ in probability.

Theorem 5 *Let H be the RKHS of an analytic and universal kernel on $\mathbb{R}^d$. Further, let $\mathcal{X} \subset \mathbb{R}^d$ be a closed ball and P be a probability measure on $\mathcal{X} \times \{\pm 1\}$ such that P_x has a density with respect to the Lebesgue measure on X and P is non-trivial. Suppose* $\sup\{\sqrt{k(x,x)} : x \in \mathcal{X}\} < \infty$. *Then for a classifier based on* (1)*, which uses a loss function of the form* (7)*, and a regularization sequence which tends to* 0 *sufficiently slowly, we have*

$$\frac{\#SV(f_{T,\lambda_n})}{n} \to \mathbb{E}_x G(\eta(x))$$

in probability.

Proof Let us fix an $\varepsilon > 0$. The proof will proceed in four steps of which the last two simply involve relating empirical averages to expectations.

Step 1. In this step we show that $f_{P,\lambda_n}(x)$ is not too close to $\pm t_0$ for most values of x. We also ensure that $f_{T,\lambda_n}(x)$ is sufficiently close to $f_{P,\lambda_n}(x)$ provided $\lambda_n \to 0$ slowly. Since $f_{P,\lambda}$ is an analytic function, for any constant c, we have

$$P_x(\{x \in X : f_{P,\lambda}(x) = c\}) > 0 \ \Rightarrow \ f(x) = c \ P_x\text{-a.s.} \tag{11}$$

Assume that $P_x(\{x \in \mathcal{X} : f_{P,\lambda}(x) = t_0\}) > 0$. By (11), we get $P_x(\{x \in \mathcal{X} : f_{P,\lambda}(x) = t_0\}) = 1$. But for small enough λ, $f_{P,\lambda} \neq t_0$ since $R_{\phi,P}(f_{P,\lambda}) \to R_{\phi,P}$ and $R_{\phi,P}(t_0) \neq R_{\phi,P}$ by the non-triviality of P. Therefore, assume that for all sufficiently large n, we have

$$P_x(\{x \in \mathcal{X} : f_{P,\lambda_n}(x) = t_0\}) = 0 \ .$$

Repeating the reasoning for $-t_0$ gives us

$$P_x(\{x \in \mathcal{X} : \left|f_{P,\lambda_n}(x) - t_0\right| \leq \delta\}) \downarrow 0 \text{ as } \delta \downarrow 0 \ ,$$

$$P_x(\{x \in \mathcal{X} : \left|f_{P,\lambda_n}(x) + t_0\right| \leq \delta\}) \downarrow 0 \text{ as } \delta \downarrow 0 \ .$$

Define the set $A_\delta(\lambda) = \{x \in \mathcal{X} : \left|f_{P,\lambda}(x) - t_0\right| \leq \delta \text{ or } \left|f_{P,\lambda}(x) + t_0\right| \leq \delta\}$. For small enough λ and for all $\varepsilon' > 0$, there exists $\delta > 0$ such that $P_x(A_\delta(\lambda)) \leq \varepsilon'$. Therefore, we can define

$$\delta(\lambda) = \frac{1}{2}\sup\{\delta > 0 : P_x(A_\delta(\lambda)) \leq \varepsilon\} \ .$$

Let $m(\lambda) = \inf\{\delta(\lambda') : \lambda' \geq \lambda\}$ be a decreasing version of $\delta(\lambda)$. Using Proposition 33 from Steinwart (2003) with $\varepsilon = m(\lambda_n)/K$, we conclude that for a sequence $\lambda_n \to 0$ sufficiently slowly, the probability of a training set T such that

$$\|f_{T,\lambda_n} - f_{P,\lambda_n}\| < m(\lambda_n)/K \tag{12}$$

converges to 1 as $n \to \infty$. It is important to note that we can draw this conclusion because $m(\lambda) > 0$ for $\lambda > 0$ (See proof of Theorem 3.5 in Steinwart, 2004). We now relate the ∞-norm of an f to its 2-norm.

$$\begin{aligned} f(x) &= \langle k(x,\cdot), f(\cdot)\rangle \leq \|k(x,\cdot)\|\,\|f\| \\ &= \sqrt{\langle k(x,\cdot), k(x,\cdot)\rangle}\|f\| \\ &= \sqrt{k(x,x)}\|f\| \leq K\|f\| \ . \end{aligned} \tag{13}$$

Thus, (12) gives us

$$\|f_{T,\lambda_n} - f_{P,\lambda_n}\|_\infty < m(\lambda_n) \ . \tag{14}$$

Step 2. In the second step, we relate the fraction of support vectors to an empirical average. Suppose that, in addition to (14), our training set T satisfies

$$\lambda_n\|f_{T,\lambda_n}\|^2 + R_{\phi,P}(f_{T,\lambda_n}) \leq R_{\phi,P} + \varepsilon \ , \tag{15}$$

$$\left|\{i : x_i \in A_{\delta(\lambda_n)}\}\right| \leq 2\varepsilon n \ . \tag{16}$$

The probability of such a T also converges to 1. For (15), see the proof of Theorem 3.5 in Steinwart (2005). Since $P_x(A_{\delta(\lambda_n)}) \leq \varepsilon$, (16) follows from Hoeffding's inequality. By definition of $R_{\phi,P}$, we have $R_{\phi,P} \leq R_{\phi,P}(f_{T,\lambda_n})$. Thus, (15) gives us $\lambda_n\|f_{T,\lambda_n}\|^2 \leq \varepsilon$. Now we use (10) to get

$$\left|\sum_{i=1}^n \alpha_i^* t_0 - \sum_{i=1}^n \alpha_i^* \xi_i^*\right| \leq 2\varepsilon \ . \tag{17}$$

Define three disjoint sets: $A = \{i : \xi_i^* < 0\}, B = \{i : \xi_i^* = 0\}$ and $C = \{i : \xi_i^* > 0\}$. We now show that B contains few elements. If x_i is such that $i \in B$ then $\xi_i^* = 0$ and we have $y_i f_{T,\lambda_n}(x_i) = t_0 \Rightarrow f_{T,\lambda_n}(x_i) = \pm t_0$. On the other hand, if $x_i \notin A_{\delta(\lambda_n)}$ then $\min\{|f_{P,\lambda_n}(x_i) - t_0|, |f_{P,\lambda_n}(x_i) + t_0|\} > \delta(\lambda_n) \geq m(\lambda_n)$, and hence, by (14), $f_{T,\lambda_n}(x_i) \neq \pm t_0$. Thus we can have at most $2\epsilon n$ elements in the set B by (16). Equation (9) gives us a bound on α_i^* for $i \in B$ and therefore

$$\left| \sum_{i \in B} \alpha_i^* t_0 \right| \leq 2\epsilon n \times h'(0) t_0 / n = 2h'(0) t_0 \epsilon \ . \tag{18}$$

Using (9), we get $\alpha_i = 0$ for $i \in A$. By definition of B, $\xi_i^* = 0$ for $i \in B$. Therefore, (17) and (18) give us

$$\left| \sum_{i \in C} \alpha_i^* t_0 - \sum_{i \in C} \alpha_i^* \xi_i^* \right| \leq 2(1 + h'(0) t_0) \epsilon = c_1 \epsilon \ .$$

where $c_1 = 2(1 + h'(0) t_0)$ is just a constant. We use (9) once again to write α_i^* as $h'(\xi_i^*)/n$ for $i \in C$:

$$\left| \frac{1}{n} \sum_{i \in C} h'(\xi_i^*) t_0 - \frac{1}{n} \sum_{i \in C} h'(\xi_i^*) \xi_i^* \right| < c_1 \epsilon \ . \tag{19}$$

Denote the cardinality of the sets B and C by N_B and N_C respectively. Then we have $N_C \leq \#SV(f_{T,\lambda_n}) \leq N_C + N_B$. But we showed that $N_B \leq 2\epsilon n$ and therefore

$$\frac{N_C}{n} \leq \frac{\#SV(f_{T,\lambda_n})}{n} \leq \frac{N_C}{n} + 2\epsilon \ . \tag{20}$$

Observe that $(\xi_i^*)_+ = 0$ for $i \in A \cup B$ and $(\xi_i^*)_+ = \xi_i^*$ for $i \in C$. Thus, we can extend the sums in (19) to the whole training set.

$$\left| \frac{1}{n} \sum_{i=1}^{n} h'((\xi_i^*)_+) t_0 - (n - N_C) \frac{h'(0) t_0}{n} - \frac{1}{n} \sum_{i=1}^{n} h'((\xi_i^*)_+)(\xi_i^*)_+ \right| < c_1 \epsilon \ .$$

Now let $c_2 = c_1 / h'(0) t_0$ and rearrange the above sum to get

$$\left| \frac{N_C}{n} - \frac{1}{n} \sum_{i=1}^{n} \left(1 - \frac{h'((\xi_i^*)_+) t_0 - h'((\xi_i^*)_+)(\xi_i^*)_+}{h'(0) t_0} \right) \right| \leq c_2 \epsilon \ . \tag{21}$$

Define $g(t)$ as

$$g(t) = 1 - \frac{h'((t_0 - t)_+) t_0 - h'((t_0 - t)_+)(t_0 - t)_+}{h'(0) t_0} \ .$$

Now (21) can be written as

$$\left| \frac{N_C}{n} - \mathbb{E}_T g(y f_{T,\lambda_n}(x)) \right| \leq c_2 \epsilon \ . \tag{22}$$

Step 3. We will now show that the empirical average of $g(yf_{T,\lambda_n}(x))$ is close to its expectation. We can bound the norm of f_{T,λ_n} as follows. The optimum value for the objective function in (1) is upper bounded by the value it attains at $f = 0$. Therefore,

$$\lambda_n \|f_{T,\lambda_n}\|^2 + R_{\phi,T}(f_{T,\lambda_n}) \leq \lambda_n.0^2 + R_{\phi,T}(0) = \phi(0) = h(t_0)$$

which, together with (13), implies that

$$\begin{aligned} \|f_{T,\lambda_n}\| &\leq \sqrt{\frac{h(t_0)}{\lambda_n}} \,, \\ \|f_{T,\lambda_n}\|_\infty &\leq K\sqrt{\frac{h(t_0)}{\lambda_n}} \,. \end{aligned} \tag{23}$$

Let $\mathcal{F}_{\lambda_n}$ be the class of functions with norm bounded by $\sqrt{h(t_0)/\lambda_n}$. The covering number in 2-norm of the class satisfies (see, for example, Definition 1 and Corollary 3 in Zhang (2002)):

$$\mathcal{N}_2(\mathcal{F}_{\lambda_n}, \varepsilon, n) \leq e^{\frac{Kh(t_0)}{\lambda_n \varepsilon^2} \log(2n+1)} \,. \tag{24}$$

Define $L_g(\lambda_n)$ as

$$L_g(\lambda_n) = \sup\left\{ \frac{|g(t) - g(t')|}{|t - t'|} : t, t' \in \left[-K\sqrt{\frac{h(t_0)}{\lambda_n}}, +K\sqrt{\frac{h(t_0)}{\lambda_n}} \right], t \neq t' \right\} \,. \tag{25}$$

Let $\mathcal{G}_{\lambda_n} = \{(x,y) \mapsto g(yf(x)) : f \in \mathcal{F}_{\lambda_n}\}$. We can express the covering numbers of this class in terms of those of $\mathcal{F}_{\lambda_n}$ (see, for example, Lemma 14.13 on p. 206 in Anthony and Bartlett (1999)):

$$\mathcal{N}_2(\mathcal{G}_{\lambda_n}, \varepsilon, n) \leq \mathcal{N}_2(\mathcal{F}_{\lambda_n}, \varepsilon / L_g(\lambda_n), n) \,. \tag{26}$$

Now, using a result of Pollard (see Pollard, 1984, Section II.6, p. 30) and the fact that 1-norm covering numbers are bounded above by 2-norm covering numbers, we get

$$P^n \left(T \in (\mathcal{X} \times \mathcal{Y})^n : \sup_{\tilde{g} \in \mathcal{G}_{\lambda_n}} |\mathbb{E}_T \tilde{g}(x,y) - \mathbb{E}_P \tilde{g}(x,y)| > \varepsilon \right) \\ \leq 8\mathcal{N}_2(\mathcal{G}_{\lambda_n}, \varepsilon/8, n) e^{-n\varepsilon^2 \lambda_n / 512 L_g^2(\lambda_n) K^2 h(t_0)} \,.$$

The estimates (24) and (26) imply that if

$$\frac{n\lambda_n^2}{L_g^4(\lambda_n) \log(2n+1)} \to \infty \text{ as } n \to \infty$$

then the probability of a training set which satisfies

$$\left| \mathbb{E}_T g(yf_{T,\lambda_n}(x)) - \mathbb{E}_P g(yf_{T,\lambda_n}(x)) \right| \leq \varepsilon \tag{27}$$

tends to 1 as $n \to \infty$.

Step 4. The last step in the proof is to show that $\mathbb{E}_P g(y f_{T,\lambda_n}(x))$ is close to $E_x G(\eta(x))$ for large enough n. Write $\mathbb{E}_P g(y f_{T,\lambda_n}(x))$ as

$$\mathbb{E}_P g(y f_{T,\lambda_n}(x)) = \mathbb{E}_x[\eta(x) g(f_{T,\lambda_n}(x)) + (1-\eta(x)) g(-f_{T,\lambda_n}(x))] \ .$$

Note that if $t^* \in F_\phi^*(\eta)$ then

$$\eta g(t^*) + (1-\eta) g(-t^*) = G(\eta) \ . \tag{28}$$

Let us verify this for three separate cases. First, when $\eta \in (0, 1-\gamma_0] \cup [\gamma_0, 1)$ the only possible values for t^* are t_0 or $-t_0$ (by Theorem 2, Part 1). Since, $g(t_0) = 0$ and $g(-t_0) = 1/(1-\gamma_0)$ the equality holds. Second, when $\eta = 1$ (the argument for $\eta = 0$ is similar), the left hand side is $g(t^*)$. In this case $t^* \geq t_0$, but the definition of g implies that $g(t) = 0$ for all $t \geq t_0$. Since $G(1) = 0$, the equality holds in this case too. Lastly, for $\eta \in (\gamma_0, 1-\gamma_0)$ we have

$$\eta g(t^*) + (1-\eta) g(-t^*) = 1 - \frac{t^*}{t_0 h'(0)} \left(\eta h'(t_0 - t^*) - (1-\eta) h'(t_0 + t^*) \right) \ .$$

Since t^* minimizes $\eta h(t_0 - t) + (1-\eta) h(t_0 + t)$ and h is differentiable, we have $\eta h'(t_0 - t^*) - (1-\eta) h'(t_0 + t^*) = 0$. Thus, we have verified (28) for all $\eta \in [0,1]$.

Define the sets $E_n = \{x \in \mathcal{X} : \rho(f_{T,\lambda_n}(x), F_\phi^*(\eta(x)) \geq \epsilon\}$. We have $P_x(E_n) \to 0$ by (3). We now bound the difference between the two quantities of interest.

$$\begin{aligned}
&\left| \mathbb{E}_P g(y f_{T,\lambda_n}(x)) - \mathbb{E}_x G(\eta(x)) \right| \\
&= \left| \mathbb{E}_x[\eta(x) g(f_{T,\lambda_n}(x)) + (1-\eta(x)) g(-f_{T,\lambda_n}(x))] - \mathbb{E}_x G(\eta(x)) \right| \\
&\leq \mathbb{E}_x \left| \eta(x) g(f_{T,\lambda_n}(x)) + (1-\eta(x)) g(-f_{T,\lambda_n}(x)) - G(\eta(x)) \right| \\
&= I_1 + I_2 \leq |I_1| + |I_2|
\end{aligned} \tag{29}$$

where the integrals I_1 and I_2 are

$$I_1 = \int_{E_n} \eta(x) g(f_{T,\lambda_n}(x)) + (1-\eta(x)) g(-f_{T,\lambda_n}(x)) - G(\eta(x))\, dP_x \ ,$$

$$I_2 = \int_{\mathcal{X} \setminus E_n} \eta(x) g(f_{T,\lambda_n}(x)) + (1-\eta(x)) g(-f_{T,\lambda_n}(x)) - G(\eta(x))\, dP_x \ .$$

Using (23) and (25) we bound $|g(\pm f_{T,\lambda_n}(x))|$ by $g(0) + L_g(\lambda_n) K \sqrt{h'(t_0)/\lambda_n}$. Since $g(0) = 1$ and $|G(\eta)| \leq 1$, we have

$$|I_1| \leq \left(1 + g(0) + L_g(\lambda_n) K \sqrt{\frac{h'(t_0)}{\lambda_n}} \right) P_x(E_n) \ .$$

If $\lambda_n \to 0$ slowly enough so that $L_g(\lambda_n) P_x(E_n) / \sqrt{\lambda_n} \to 0$, then for large n, $|I_1| \leq \epsilon$. To bound $|I_2|$, observe that for $x \in \mathcal{X} \setminus E_n$, we can find a $t^* \in F_\phi^*(\eta(x))$, such that $|f_{T,\lambda_n}(x) - t^*| \leq \epsilon$. Therefore,

$$\eta(x) g(f_{T,\lambda_n}(x)) + (1-\eta(x)) g(-f_{T,\lambda_n}(x)) = \eta(x) g(t^*) + (1-\eta(x)) g(-t^*) + \Delta \ .$$

where $|\Delta| \leq c_3 \epsilon$ and the constant c_3 does not depend on λ_n. Using (28), we can now bound $|I_2|$:

$$|I_2| \leq c_3 \epsilon (1 - P_x(E_n)) \leq c_3 \epsilon \ .$$

We now use (29) to get

$$\left| \mathbb{E}_P g(y f_{T,\lambda_n}(x)) - \mathbb{E}_x G(\eta(x)) \right| \leq (c_3 + 1) \epsilon \ . \tag{30}$$

Finally, combining (20), (22), (27) and (30) proves the theorem. ■

5. Conclusion

We saw that the decision functions obtained using minimization of regularized empirical ϕ-risk approach $F^*_\phi(\eta(\cdot))$. It is not possible to preserve sparseness on intervals where $F^*_\phi(\cdot)$ is invertible. For the regions outside that interval, sparseness is maintained to some extent. For many convex loss functions, the general lower bounds known previously turned out to be quite loose.

But that leaves open the possibility that the previously known lower bounds are actually achievable by some loss function lying outside the class of loss functions we considered. However, we conjecture that it is not possible. Note that the right hand side of Theorem 5 only depends on the left derivative of the loss function at t_0 and the right derivative at $-t_0$. The derivatives at other points do not affect the asymptotic number of support vectors. This suggests that the assumption of the differentiability of ϕ before the point where it attains its minimum can be relaxed. It may be that results on the continuity of solution sets of convex optimization problems can be applied here (e.g., see Fiacco, 1983).

Acknowledgments

This work was supported in part by ARO under MURI grant DAAD 190210383. Thanks to Grace Wahba and Laurent El Ghaoui for helpful discussions.

References

Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, Cambridge, 1999.

Peter L. Bartlett, Michael I. Jordan, and Jon D. McAuliffe. Large margin classifiers: Convex loss, low noise and convergence rates. In *Advances in Neural Information Processing Systems* **16**. MIT Press, 2004.

Anthony V. Fiacco. *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming*. Academic Press, New York, 1983.

Gábor Lugosi and Nicolas Vayatis. On the Bayes-risk consistency of regularized boosting methods. *Annals of Statistics*, 32(1):30–55, 2004.

David Pollard. *Convergence of Stochastic Processes*. Springer-Verlag, New York, 1984.

R Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, 1970.

Ingo Steinwart. Sparseness of support vector machines. *Journal of Machine Learning Research*, 4: 1071–1105, 2003.

Ingo Steinwart. Sparseness of support vector machines – some asymptotically sharp bounds. In *Advances in Neural Information Processing Systems* **16**. MIT Press, 2004.

Ingo Steinwart. Consistency of support vector machines and other regularized kernel classifiers. *IEEE Transactions on Information Theory*, 51(1):128–142, 2005.

Grace Wahba. Soft and hard classification by reproducing kernel Hilbert space methods. *Proceedings of the National Academy of Sciences USA*, 99(26):16524–16530, 2002.

Tong Zhang. Covering number bounds of certain regularized linear function classes. *Journal of Machine Learning Research*, 2:527–550, 2002.

Tong Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals of Statistics*, 32(1):56–85, 2004.

Journal of Machine Learning Research 8 (2007) 791-812 Submitted 9/05; Revised 12/06; Published 4/07

Concave Learners for Rankboost

Ofer Melnik MELNIK@DIMACS.RUTGERS.EDU
DIMACS
Rutgers University
Piscataway, NJ 08854 USA

Yehuda Vardi VARDI@STAT.RUTGERS.EDU
Cun-Hui Zhang CZHANG@STAT.RUTGERS.EDU
Department of Statistics
Rutgers University
Piscataway, NJ 08854 USA

Editor: Donald Geman

Abstract

Rankboost has been shown to be an effective algorithm for combining ranks. However, its ability to generalize well and not overfit is directly related to the choice of weak learner, in the sense that regularization of the rank function is due to the regularization properties of its weak learners. We present a regularization property called *consistency in preference and confidence* that mathematically translates into monotonic concavity, and describe a new weak ranking learner (MWGR) that generates ranking functions with this property. In experiments combining ranks from multiple face recognition algorithms and an experiment combining text information retrieval systems, rank functions using MWGR proved superior to binary weak learners.

Keywords: rankboost, ranking, convex/concave, regularization

1. Ranking Problems

A ranking problem is a learning problem that involves ranks as the inputs, the outputs or both. An example where ranks are used as inputs is a collaborative filtering application where people are asked to rank movies according to their preferences. In such an application the ranks assigned by different people are combined to generate recommendations. Another type of problem in which ranks are used as inputs are meta-search problems, where the ranks of multiple search engines are combined (Dwork et al., 2001). However, the inputs to a ranking problem are not always ranks. An object recognition ranking problem (Kittler and Roli, 2000) may receive as inputs a graphical representation and output a ranking of the possible objects, sorted by likelihood.

The outputs of a ranking problem may also be ranks. For example, in combining multiple search engines the output are ranks which are a synthesis of the ranks from the individual search engines. A similar meta-recognition problem is the combination of the outputs of multiple face- recognition systems to improve the accuracy of detecting the correct face. While the inputs and outputs of this problem are ranks, the outputs can be simplified to only return the most likely candidate. Another example where the outputs do not need to be complete ranks could be an information retrieval combination task. In such a task the inputs might be ranks of sets of documents with respect to a particular query by different experts. Again, the outputs could be the complete ranks, or more simply a designation for each document of whether it is relevant or not to the particular query.

2. Rankboost

The rankboost algorithm (Freund et al., 2003) tries to address this variety in ranking problems by maintaining generality in how it regards its inputs and how it applies different loss functions to outputs. The rankboost algorithm works on *instances* which are the discrete items (e.g., movies or faces) that either are ranked as input or are to be ranked as output. To allow a general input mechanism, the inputs to rankboost are called the *ranking features* of an instance, specified as $f_j(x)$ which is the j-th ranking feature of instance x.

While this generality in how inputs are handled is potentially powerful, in the original rankboost paper as well as in this paper, only the case where the inputs are different rankings of the instances is considered. Thus, in this paper, the inputs to rankboost are constituent ranks, denoted as $f_1 \dots f_n$, where $f_j(x') < f_j(x'')$ implies that *instance* x' has a better rank than instance x'' under ranking f_j. For example, in some of the experiments we present, each f_j is the ranking result of a particular face recognition algorithm.

The output of rankboost is a new *ranking function*, $H(x)$, which defines a linear ordering on the instances, that is, $H(x') < H(x'')$ iff x' has a better rank than x''. In rankboost

$$H(x) = \sum_{t=1}^{T} w_t h_t(x), \tag{1}$$

a weighted sum of weak ranking learners, where the $h_t(x)$'s are relatively simple learned functions of the constituent rankings.

To address the variety in possible loss functions of the outputs, in rankboost the desirable properties for the output loss function are specified with a *favor function*, $\Phi : X \times X \to \mathbb{R}$, where X is the space of instances (note that this function has been renamed from "preference function" to avoid confusion with the use of preference in this paper in Section 4). Here $\Phi(x',x'') > 0$ means that x'' should be better ranked than x' for a given query. It is an anti-symmetric function, that is, $\Phi(x',x'') = -\Phi(x'',x')$ and $\Phi(x,x) = 0$, which avoids loops where two instances should both be ranked better than the other. Also $\Phi(x',x'') = 0$ when there is no favor in how two instances should be relatively ranked. For example, as described in Section 6.1, for the face recognition combination problem described above the favor function can be used to specify that the correct identity should be given a better rank than all other identities, while zeroing all other entries in the favor function, giving no favor in how incorrect identities are ranked between them. In a similar fashion for the information retrieval combination task mentioned above, the favor function can be specified such that all relevant documents should be better ranked than irrelevant documents, without specifying favor for the ordering between relevant documents and the ordering between irrelevant documents (Section 7.1).

Rankboost is shown in Algorithm 1 as described in Freund et al. (2003). It uses the favor function to specify an initial weight function over instance pair orderings:

$$D(x',x'') = c \cdot \max\left(0, \Phi\left(x',x''\right)\right), \tag{2}$$

where $c = \left[\sum_{x',x''} \max\left(0, \Phi\left(x',x''\right)\right)\right]^{-1}$. The algorithm itself is very similar to adaboost (Freund and Schapire, 1996). At each iteration the algorithm selects the weak learner that best maximizes a

Algorithm 1 rankboost algorithm for generating a ranking function.

```
Input: constituent ranks, a favor function, and a class of weak learners with outputs between 0
and 1 and an appropriate training algorithm.
Output: ranking function $H(x)$ (Eq. 1)
Initialize $D^{(1)} = D$ (Eq. 2)
for $t = 1 \ldots T$ do
    Find weak learner, $h_t$, that maximizes $r(h) = \sum_{x',x''} D^{(t)}(x',x'')(h(x') - h(x''))$ (Eq. 4).
    Choose $w_t = 0.5 \ln((1 + r(h_t)) / (1 - r(h_t)))$. (Eq. 5)
    Update:
    $D^{(t+1)}(x',x'') = D^{(t)}(x',x'') \exp(w_t (h_t(x') - h_t(x''))) / Z_t$
    where $Z_t$ is chosen to make $\sum_{x',x''} D^{(t+1)}(x',x'') = 1$
end for
```

rank function's utility, $r(h)$, (Eq. 4). It then assigns it a weight in proportion to its performance and adds the weighted weak learner to the ranking function, $H(x)$. After which the weight function D is adjusted to reflect the impact of the weak learner. Freund et al. (2003) prove that the rankboost algorithm iteratively minimizes the ranking loss function, a measure of the quantity of misranked pairs:

$$\sum_{x',x''} D\left(x',x''\right) I\left[H\left(x'\right) \leq H\left(x''\right)\right]$$

where I is the indicator function (1 if true, 0 otherwise) and $H(x)$ is a ranking function output by rankboost. The rankboost paper (Freund et al., 2003) uses *binary weak learners* of the following functional form:

$$h(x) = \begin{cases} 1 & if\, f_j(x) > \theta, \\ 0 & if\, f_j(x) \leq \theta, \\ q_{def} & if\, f_j(x)\, undefined. \end{cases} \tag{3}$$

Each binary weak learner, h, operates on a single f_j (ranking feature), giving a binary output of 0 or 1 depending on whether the instance's rank is smaller than a learned parameter, θ. Note that function $h(x)$ in (3), which is dependent on only one $f_j(x)$ with a fixed j, is monotonic increasing but not convex or concave. If there is no rank for the instance then it returns a prespecified q_{def} value.

3. Rankboost, the Weak Learner and Regularization

While rankboost inherits the accuracy of adaboost and has been shown to be very successful in practice, in two important ways it is very different from adaboost and similar classifier leveraging algorithms (Meir and Ratsch, 2003). The first is the choice of the weak learner, h. In adaboost the weak learner is expected to minimize the weighted empirical classification error:

$$\sum_{i=1}^{N} d^{(t)}(i) I\left[y_i \neq h(z_i)\right],$$

where y_i is the class label, I is the indicator function and $d^{(t)}$ is a weighting over training samples. This is a standard function to minimize in classification with many possible types of algorithms to choose from as possible weak learners. In contrast the weak ranking learner for rankboost (with outputs in $[0,1]$) needs to maximize the following rank utility function:

$$r = r(h) = \sum_{x',x''} D^{(t)}(x',x'')(h(x') - h(x'')), \tag{4}$$

where $D^{(t)}$ is the weight function over pairs of instances. As Eq. 4 contains the term $h(x') - h(x'')$, short of linear learners this equation can not be concave in h or easily approximated by a concave function. Therefore the weak learner needs to be optimized either by heuristics or by brute force, which limits the choice of h. It is not surprising that the original rankboost paper only included one type of weak learner, a binary threshold function (Eq. 3) that was tuned using brute force.

The second difference between rankboost and adaboost also concerns the weak ranking learner. One feature of boosting that has sparked a great deal of interesting research is the algorithm's ability to avoid overfitting for low noise classification problems (with modifications to higher noise problems), see Meir and Ratsch (2003) for a survey. In contrast for rankboost it is only by limiting the type of underlying weak learners that overfitting is avoided. In their paper, Freund et al. (2003) show that not using weak ranking learners with cumulative positive coefficients leads to overfitting and poor performance quite quickly. Therefore, choosing a weak learner that regularizes the ranking function, the output of rankboost, is very important for achieving accuracy and avoiding overfitting.

It is clear from the above discussion that the choice of a weak ranking learner for rankboost is important and non trivial. First, the learner must be efficiently tunable with respect to Eq. 4, typically limiting its complexity. Second, the learner itself must demonstrate regularization properties that are appropriate for ranking functions.

In this paper we present a new weak ranking learner that enforces *consistency in preference and confidence* for the ranking function by being monotonic and concave. We start with a discussion of these regularization properties, theoretically justify them, and show what they mean in terms of the final ranking function. Then we present a new learner, *Minimum Weighted Group Ranks* (MWGR), that satisfies these properties and can be readily learned. This learner is tested and compared with the binary learner of rankboost on combining multiple face recognition systems from the FERET study (Phillips et al., 2000) and on an information retrieval combination task from TREC (Voorhees and Harman, 2001).

4. Regularizing Ranking Functions with Consistency in Preference and Confidence

In this paper, as Freund et al. (2003), we consider ranking functions $H(x)$ which depend on x only through the values of the ranking features, $y_j = f_j(x)$, for that instance, so that $H(x) = G(f_1(x), \dots, f_n(x))$, for certain functions $G(y_1, \dots, y_n) = G(\mathbf{y})$. Here, we assume that the $f_j(x)$ is an actual rank assigned to an instance, x, by the j-th ranker. Note that if the original data are numerical scores then they can easily be converted to rankings. Freund et al. (2003) make a strong case for conversion of raw measures to relative orderings (rankings) over combining measures directly, arguing that it is more general and avoids the semantics of particular measures.

As the y_j's are ranks instead of points in a general space, care should be taken as to the functional form of G. A great deal of literature in social choice theory (Arrow, 1951) revolves around the properties of various rank combination strategies that try to achieve fair rankings. In this machine learning case our motivations are different. Fairness is not the goal; the goal is to improve the accuracy or performance of the ranking function. Thus, regularization, by functionally constraining G, is used to confer information on how to interpret ranks in order to ultimately improve accuracy.

Freund et al. (2003) imposed the regularization principle of *monotonicity* on the ranking function. Monotonicity encompasses a belief that for individual features a smaller rank is always considered better than a bigger rank. It means that for every two rank vectors, $\mathbf{a}$ and $\mathbf{b}$, if $a_j \leq b_j$, $j = 1,\ldots,n$ then $G(\mathbf{a}) \leq G(\mathbf{b})$. Monotonicity was enforced by requiring that the coefficients, w_t in Eq. 1, combining the binary weak learners (Eq. 3) were cumulatively positive. As shown by Freund et al. (2003), without enforcing monotonicity the rankboost algorithm quickly overfits.

In this section we present another regularization principle, consistency in preference and confidence (which includes monotonicity). A ranking function with this regularization property should be consistent in its preference for the individual rankers and also in how it captures their confidence. The following example illustrates these two concepts.

4.1 Grocer Example

A grocer needs to make 2 decisions, to decide between stocking oat bran vs. granola and to decide between stocking turnips vs. radishes. The grocer asks her consultants to express their opinion about stocking each item, and based on their responses makes her 2 decisions.

First of all, in either problem, the grocer will adopt the opinion of her consultants if they all agree with each other, that is, they all prefer granola over oat bran in the first problem.

Lets assume the grocer considered the first problem and chose granola over oat bran. What this implies is that the grocer adopted the opinions of the consultants that preferred granola over oat bran.

Now consider the turnips vs. radishes decision. Lets say that the same consultants that liked granola more also liked radishes more (and the same ones that like oat bran more like turnips more). Also, for this decision the radish lovers actually feel more confident in their choice than they did for granola, while the turnip lovers are more unsure than they were for oat bran.

Then for this second choice, if the grocer is consistent in her method of combining the opinions of her consultants, we would expect her to pick radishes over the turnips. In addition, we would expect her to be more confident in this second decision as a reflection of the consultants relative confidence. The above properties of preference and confidence imply that preference should be applied consistently across different problems and confidence should reflect the relative confidences of the individual consultants.

4.2 The General Principle of Consistency in Preference and Confidence

To generalize the above grocer's problem consider the problem of combining the opinions of a panel of consultants on several multiple-choice decision problems. Suppose for each decision problem each consultant provides his preference as one of the possible choices and a qualitative measure of the confidence level for his preference. The consistency principle in preference and confidence holds if the combiner always agrees with at least one of the consultants in each decision problem and that the following condition holds for any pair of decision problems.

Definition 1. Consistency principle for a pair of decision problems: *Suppose that for the first decision problem, the combiner adopts the preference of a subset A of consultants in the sense that A is the (non empty) set of all consultants whose preference is identical to that of the combiner.*

Suppose that for the second decision problem, the preference of the consultants in A is again identical. Suppose further that compared with the confidence level for the first decision problem, each consultant in A has higher or the same confidence level for his preference in the second problem, while each consultant with a different preference than A for the second problem has lower or the same confidence level. Then, the preference of the combiner for the second problem is identical to that of the consultants in A, and the confidence level of the combiner for the second problem is at least as high as his confidence level for the first problem.

Let B be the set of consultants whose preferences are different from that of the combiner in the first decision problem, that is, $B = A^c$. If some members of B switch sides or have no preference in the second problem, the combiner would be even more confident about the adoption of the preference of A in the second problem regardless of the confidence of those who switch sides. Thus, Definition 1 requires only those against the preference of A in the second problem (necessarily a subset of B since members of A act in unison in both problems) to have lower or equal confidence in the second problem, instead of all members of B. This is taken into consideration in Theorem 1 below.

Note that while preference for individual consultants is specified within each decision problem, two decision problems are needed to compare the qualitative measure of confidence. However, comparison of confidence is not always possible (it is a partial ordering). In particular, the level of confidence between different experts may not be comparable, and the levels of confidence of a given expert (or the combiner) for different decision problems are not always comparable.

4.3 Confidence for Ranks and Ranking Functions

In order to apply consistency to rank combination we need to specify what we mean by more or less confidence. For ranks we make the assumption that a constant change of ranks requires more confidence for low ranks than high ranks. For example, we would expect the difference between ranks of 1 and 3 to represent a more significant distinction on the part of a ranker than would for example the difference between ranks 34 and 36 which may be almost arbitrarily assigned. Specifically we make the following definition:

Definition 2. Preference and confidence for univariate rank values *For any pair of rank values $\{r, r'\} \subset \mathbf{R}$ with $r < r'$, by monotonicity r is preferred. For any two pairs of rank values $\{r, r'\}$ and $\{r'', r'''\}$ with $r < r'$ and $r'' \leq r'''$, the confidence level for $\{r, r'\}$is higher than that of $\{r'', r'''\}$ if*

$$r \leq r'', \ r''' - r'' \leq r' - r$$

with at least one inequality. The pair$\{r, r'\}$provides no preference if $r = r'$. Likewise, if either $r' - r = r''' - r'' = 0$ or $r'' - r = r''' - r' = 0$, the two pairs $\{r, r'\}$ and $\{r'', r'''\}$are defined to have the same level of confidence.

In this definition confidence between pairs of ranks can only be compared if the pair with the lowest rank has a gap at least as large as the other pair. Thus, we are actually comparing two numbers, that is, we are doing a vector comparison. As such, this comparison does not cover all pairs of ranks and applies only a partial ordering on rank pairs. As a regularization principle, a partial ordering is desirable since it does not overly constrain the ranking function and allows for flexibility.

As the rank function, G, is a univariate function, we apply the same definitions of preference and confidence to it. That is, for a ranking function G and for any fixed rank vectors, $\mathbf{y}$ and $\mathbf{y}'$, the preferred rank vector is the one with smaller G, that is, $\mathbf{y}$ is preferred iff $G(\mathbf{y}) < G(\mathbf{y}')$. For confidence again we need to consider pairs of rank vectors, $\{\mathbf{y},\mathbf{y}'\}$ and $\{\mathbf{y}'',\mathbf{y}'''\}$, with $G(\mathbf{y}) \leq G(\mathbf{y}')$ and $G(\mathbf{y}'') \leq G(\mathbf{y}''')$. If $G(\mathbf{y}) \leq G(\mathbf{y}'')$ and $G(\mathbf{y}') - G(\mathbf{y}) \geq G(\mathbf{y}''') - G(\mathbf{y}'')$ then we say that the first decision, between $\mathbf{y}$and $\mathbf{y}'$, shows equal or more confidence than the second decision between $\mathbf{y}''$ and $\mathbf{y}'''$.

This numerical method of capturing confidence and preference in ranks, $\mathbf{y}$, and ranking functions, G, allows us to apply Definition 1. Specifically, for a pair of binary decisions the confidence of a consistent ranking function increases for the second decision if in the second decision the confidence of the agreeing rank values are comparable and increase and the confidence of the disagreeing rank values are comparable and decrease, with the exception of those that switched sides.

4.4 Three-Point Consistency in Preference and Confidence for Combining Ranks

As described, the consistency property is over two binary decision problems. In this section we consider ranking functions, G, that have consistency in preference and confidence for all pairs of binary decision problems involving three rank vectors $\mathbf{y}$, $\mathbf{y}'$ and $\mathbf{y}''$. We show that such functions have specific mathematical properties under this three-point consistency principle in preference and confidence for ranks..

Theorem 1: *For a ranking function G whose domain is convex, three-point consistency in preference and confidence holds for G iff $G(\mathbf{y})$ is nondecreasing in individual components of $\mathbf{y}$ and is jointly concave in $\mathbf{y}$.*

Proof of Necessity: Assume that the ranking function G exhibits three-point consistency in preference and confidence for any three rank vectors.

If $\mathbf{y} \leq \mathbf{y}'$ component wise, then the individual components of $\mathbf{y}$ all agree in their preference to $\mathbf{y}$, so that $G(\mathbf{y}) \leq G(\mathbf{y}')$ by the consistency of G in preference. This implies the monotonicity of G in $\mathbf{y}$.

We pick three points $\mathbf{y}$, $\mathbf{y}-\mathbf{a}$ and $\mathbf{y}+\mathbf{a}$, such that all points are rank vectors. Consider the pair of binary comparison problems, $\mathbf{y}$ vs. $\mathbf{y}+\mathbf{a}$ and $\mathbf{y}-\mathbf{a}$ vs. $\mathbf{y}$. Assume that $G(\mathbf{y}) < G(\mathbf{y}+\mathbf{a})$. These three points are comparable by Definition 2 ($r''' - r'' = r' - r$ for all components in the rank vectors). Since the agreeing rankers, $A = \left\{ j \middle| y_j < y_j + a_j \right\}$, in the $\mathbf{y}$ vs. $\mathbf{y}+\mathbf{a}$ comparison are greater than in the $\mathbf{y}-\mathbf{a}$ vs. $\mathbf{y}$ comparison, and the disagreeing ranks, outside A, are smaller, then by the consistency of the combiner in preference and confidence (definition 1), $G(\mathbf{y}-\mathbf{a}) \leq G(\mathbf{y})$ and $G(\mathbf{y}) - G(\mathbf{y}-\mathbf{a}) \geq G(\mathbf{y}+\mathbf{a}) - G(\mathbf{y})$.

A function G is concave in a convex domain iff $2G(\mathbf{y}) \geq G(\mathbf{y}-\mathbf{a}) + G(\mathbf{y}+\mathbf{a})$, for every $\mathbf{y}$ and $\mathbf{a}$ with $\mathbf{y} \pm \mathbf{a}$ in its domain. To verify this inequality for a particular $\mathbf{y}$ and $\mathbf{a}$, we must have $G(\mathbf{y}+\mathbf{a}) > G(\mathbf{y})$, $G(\mathbf{y}-\mathbf{a}) > G(\mathbf{y})$ or the third case of $G(\mathbf{y}) \geq \max\{G(\mathbf{y}+\mathbf{a}), G(\mathbf{y}-\mathbf{a})\}$. We have already proven that the consistency properties of preference and confidence imply $G(\mathbf{y}) - G(\mathbf{y}-\mathbf{a}) \geq G(\mathbf{y}+\mathbf{a}) - G(\mathbf{y})$ in the first case. By symmetry this requirement also must hold for $-\mathbf{a}$, so that $G(\mathbf{y}) - G(\mathbf{y}+\mathbf{a}) \geq G(\mathbf{y}-\mathbf{a}) - G(\mathbf{y})$ in the second case. This completes the necessity part of the proof since $2G(\mathbf{y}) \geq G(\mathbf{y}-\mathbf{a}) + G(\mathbf{y}+\mathbf{a})$ automatically holds in the third case.

Proof of Sufficiency: Assume the ranking function G is nondecreasing in individual components of $\mathbf{y}$ and jointly concave in $\mathbf{y}$. We need to prove consistency in preference and confidence for a pair

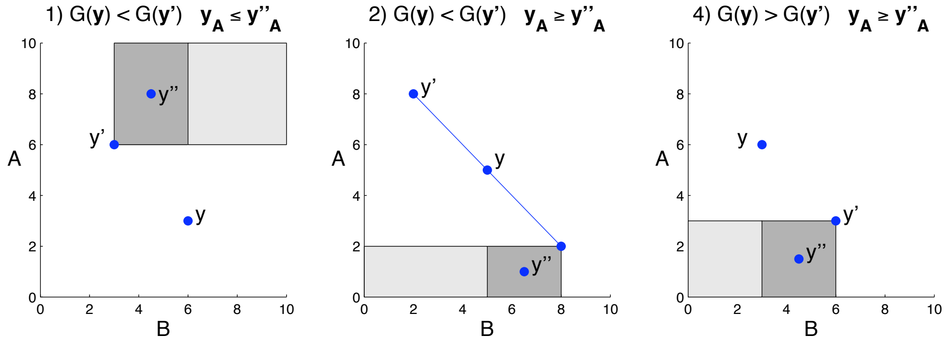


Figure 1: Consider two decision problems in two dimensions involving three points, $\mathbf{y}$, $\mathbf{y}'$ and $\mathbf{y}''$, where the first decision problem is to choose between $\mathbf{y}$ and $\mathbf{y}'$ and the second problem is to choose between $\mathbf{y}$ and $\mathbf{y}''$. The cases where we expect consistency in preference and confidence (Definitions 1 and 2) can be enumerated by the result of the first decision problem and relative location of $\mathbf{y}''$. The darker gray box is the location where B has lesser or equal confidence in the second problem, and the lighter gray box is where B switches sides.

of decision problems involving three rank vectors $\mathbf{y}$, $\mathbf{y}'$ and $\mathbf{y}''$. Without loss of generality, suppose that the first problem is to choose between $\mathbf{y}$ and $\mathbf{y}'$ and the second problem is to choose between $\mathbf{y}$ and $\mathbf{y}''$. We break the proof up by the results of the comparison between $\mathbf{y}$ vs. $\mathbf{y}'$and the relative location of the third ranking vector $\mathbf{y}''$ that satisfies the preference and confidence requirements. If $G(y) = G(y') = G(y'')$, the combiner has equal confidence in the two decision problems by definition 2, so that the consistency principle holds automatically. Thus, we only need to consider the cases where $G(y) \neq G(y')$. Figure 1 illustrates three of these cases two dimensionally, where there is a single agreeing component A, the y-axis, and a single disagreeing component B, the x-axis.

Let A be the set of agreeing indices, $A = \left\{ j : \mathrm{sgn}(y'_j - y_j) = \mathrm{sgn}\left(G(y') - G(y)\right) \neq 0 \right\}$, and $B = A^c$ the set of disagreeing indices. We use the notation $\mathbf{y_A} = (y_j,\ j \in A)$ to describe corresponding subvectors. Also, when used, vector inequalities are component wise.

Case 1: $G(\mathbf{y}) < G(\mathbf{y}')$ and $\mathbf{y_A} \leq \mathbf{y''_A}$

In the first decision problem, as G agrees with A and disagrees with B

$$\mathbf{y_A} < \mathbf{y'_A},\ \mathbf{y_B} \geq \mathbf{y'_B}.$$

We also know that $A \neq \emptyset$ since otherwise $\mathbf{y} \geq \mathbf{y}'$and therefore by monotonicity $G(\mathbf{y}) \geq G(\mathbf{y}')$, which is a contradiction.

For the second decision problem we consider the values of $\mathbf{y}''$ which are consistent with the confidence assumption (Definitions 1 and 2), that is, agreeing with more confidence along the A indices and disagreeing with less confidence or switching sides along the B indices. As $\mathbf{y_A} \leq \mathbf{y''_A}$, either by the confidence relationship or by switching sides (see Figure 1) these $\mathbf{y}''$ values satisfy

$$\mathbf{y''_A} - \mathbf{y_A} \geq \mathbf{y'_A} - \mathbf{y_A}, \quad \mathbf{y_B} - \mathbf{y''_B} \leq \mathbf{y_B} - \mathbf{y'_B}$$

which implies $\mathbf{y}' \leq \mathbf{y}''$, and therefore $G(\mathbf{y}') \leq G(\mathbf{y}'')$ by monotonicity. Thus, $G(\mathbf{y}) < G(\mathbf{y}') \leq G(\mathbf{y}'')$, which implies that in the second decision problem of choosing between $\mathbf{y}$ and $\mathbf{y}''$, the preference of G is the same as that of A (i.e., $\mathbf{y}$ since $\mathbf{y_A} \leq \mathbf{y}''_\mathbf{A}$) and the confidence of G is at least as high as the first decision problem.

Case 2: $G(\mathbf{y}) < G(\mathbf{y}')$ **and** $\mathbf{y_A} \geq \mathbf{y}''_\mathbf{A}$

Since in this case $\mathbf{y_A} \geq \mathbf{y}''_\mathbf{A}$, then either by the confidence relationship or by switching sides $\mathbf{y}''$ satisfies

$$\mathbf{y_A} - \mathbf{y}''_\mathbf{A} \geq \mathbf{y}'_\mathbf{A} - \mathbf{y_A}, \quad \mathbf{y}''_\mathbf{B} - \mathbf{y_B} \leq \mathbf{y_B} - \mathbf{y}'_\mathbf{B}.$$

This implies that $\mathbf{y}' + \mathbf{y}'' \leq 2\mathbf{y}$, which means that

$$G(\mathbf{y}') + G(\mathbf{y}'') \leq 2G\left(\left(\mathbf{y}' + \mathbf{y}''\right)/2\right) \leq 2G(\mathbf{y})$$

by the concavity and monotonicity of G. Thus, $G(\mathbf{y}) - G(\mathbf{y}'') \geq G(\mathbf{y}') - G(\mathbf{y})$ and since $G(\mathbf{y}') > G(\mathbf{y})$, we have that $G(\mathbf{y}) - G(\mathbf{y}'') > 0$ and thus $G(\mathbf{y}) > G(\mathbf{y}'')$. Therefore we see that the preference in G for the two decision problems is the same as A (with $\mathbf{y}$ in the first problem and with $\mathbf{y}''$ in the second problem) and the confidence is no smaller for the second comparison.

Case 3: $G(\mathbf{y}) > G(\mathbf{y}')$ **and** $\mathbf{y_A} \leq \mathbf{y}''_\mathbf{A}$

Since $\mathbf{y}'$ is preferred in the first decision problem we have

$$\mathbf{y_A} > \mathbf{y}'_\mathbf{A}, \mathbf{y_B} \leq \mathbf{y}'_\mathbf{B}.$$

For the confidence assumption to hold, that is, greater confidence for A in the second problem (Definition 2), the smaller ranks in the second problem have to be smaller than the smaller ranks in the first problem. But with $\mathbf{y_A} \geq \mathbf{y}'_\mathbf{A}$ and $\mathbf{y_A} \leq \mathbf{y}''_\mathbf{A}$ that can only be if $\mathbf{y_A} = \mathbf{y}'_\mathbf{A}$, which leads to $\mathbf{y} \leq \mathbf{y}'$, and by monotonicity implies $G(\mathbf{y}) \leq G(\mathbf{y}')$. However that is a contradiction to $G(\mathbf{y}) > G(\mathbf{y}')$, and therefore this is not a viable case for comparing confidence.

Case 4: $G(\mathbf{y}) > G(\mathbf{y}')$ **and** $\mathbf{y_A} \geq \mathbf{y}''_\mathbf{A}$

Since in this case $\mathbf{y_A} \geq \mathbf{y}''_\mathbf{A}$, then either by the confidence relationship or by switching sides $\mathbf{y}''$ satisfies

$$\mathbf{y_A} - \mathbf{y}''_\mathbf{A} \geq \mathbf{y}'_\mathbf{A} - \mathbf{y_A}, \quad \mathbf{y}''_\mathbf{B} - \mathbf{y_B} \leq \mathbf{y_B} - \mathbf{y}'_\mathbf{B},$$

which means that $\mathbf{y}'' \leq \mathbf{y}'$. Thus, by the monotonicity of G, $G(\mathbf{y}'') \leq G(\mathbf{y}')$. Since $G(\mathbf{y}') < G(\mathbf{y})$ the preference in the second decision problem is also with A (i.e., $\mathbf{y}''$) and the confidence is at least as high as the first decision problem.

4.5 Applying Regularization to Rankboost

It follows from the above theorem that to have consistency in preference and confidence we desire ranking functions that are monotonic and concave. In rankboost $H(x) = \sum w_t h_t(x)$ (Eq. 1). To make H an increasing and concave function of constituent rankings $y_j = f_j(x)$ we need to constrain the weak ranking learners. If the learners themselves are monotonically increasing and concave functions of y_j, then linearly combining them with positive weights will give an H that is also an increasing and concave function of y_j.

In this paper, we apply the "third method" (Freund et al., 2003) to setting a w_t weight value. That is, weak learners are selected on their ability to maximize r from Eq. 4 and then

$$w_t = 0.5 \ln\left((1 + r_{\max}) / (1 - r_{\max})\right). \tag{5}$$

Therefore, using monotonic and concave weak learners we select only ones that rankboost gives a positive r value to, which renders a positive w_t weight. If no r values are positive the rankboost algorithm stops.

We mention that a ranking function can be construed as the negative of a score function, that is, $G(\mathbf{y}) = -S(\mathbf{y})$. For score functions these regularization properties become monotonically decreasing, and convex (Melnik et al., 2004).

5. Minimum Weighted Group Ranks

The functional structure of the new learner we propose is

$$h(y) = \min\{\gamma_1 y_1, \ldots, \gamma_n y_n, 1\}, \tag{6}$$

where $\mathbf{y} = (y_1, \ldots, y_n) = (f_1(x), \ldots, f_n(x))$ the vector of ranking features, and the γ_j are learned positive coefficients. Note that the function's range is in $[0,1]$ due to the 1 term in the min function.

Using rankings as our features, the learner function (Eq. 6) is monotonically increasing. It is also a concave function in $\mathbf{y}$. Thus if these learners are linearly combined with positive weights, the resulting ranking function, H, will have three-point consistency in confidence and preference.

To gain some intuition, the functional form of the learner is related to the Highest Rank combination method (Ho, 1992) that assigns combined ranks as the best rank each class receives from any ranker. This is a confidence based approach, as Highest Rank bets on the classifier that is most confident, the one giving the best rank. As such, a single learner can potentially be error prone. But as we combine many of these learners during boosting, it becomes more powerful, allowing the confidence of different classifiers to be weighed with preference for their potential accuracy.

5.1 Learning

At each boosting iteration, rather than selecting from all possible weak learners of form (Eq. 6), we limit our choices by building new weak learners out of the ones that have been previously trained. Let $\mathcal{F} = \{f_1(x), \ldots, f_n(x)\}$ be the set of ranking features. Recall that in rankboost $H(x) = \sum_t w_t h_t(x)$, where $h_t(x) = \min\left\{\gamma_1^{(t)} f_1(x), \ldots, \gamma_n^{(t)} f_n(x), 1\right\}$ and $\gamma_j^{(t)}$ are learned coefficients. We set

$$\widetilde{\mathcal{H}}_t = \left\{\widetilde{h}(x) \,\middle|\, \widetilde{h}(x) = \min\left\{\gamma_1^{(s)} f_1(x), \ldots, \gamma_n^{(s)} f_n(x)\right\}, s \le t\right\}$$

and select $h_{t+1}(x)$ from weak learners of the form

$$h_{new}(x) = \min\left\{\alpha\widetilde{h}(x), \beta f(x), 1\right\} \tag{7}$$

with $\widetilde{h}(x) \in \widetilde{\mathcal{H}}_t$ and $f(x) \in \mathcal{F}$. This learner can be rewritten in the form of Eq. 6 with the $\gamma_j^{(t+1)}$ derived from the learned α, β, $\widetilde{h}(x)$ and $f(x)$. Thus, at each iteration we look at combinations of the features and existing learners. As discussed in the next section, we can either consider all such combinations, or use a heuristic selection strategy.

We propose to optimize α and β separately, in stages. That is, given a value for one of the variables we optimize the other. As α and β are symmetric we show how to optimize β given α.

We need to find a value of β that maximizes r in Eq. 4. Freund et al. (2003) pointed out that this equation can be rewritten as:

$$\begin{aligned} r &= \sum_{x',x''} D^{(t)}(x',x'')(h(x')-h(x'')) \\ &= \sum_{x} \pi(x)h(x) \end{aligned}$$

where $\pi(x) = \sum_{x'} \left(D^{(t)}(x',x) - D^{(t)}(x,x')\right)$. Given the form of Eq. 7 we can write r as a function of β,

$$\begin{aligned} r(\beta) &= \sum_{\beta f_j(x) \le \min\left(\alpha\widetilde{h}(x),1\right)} (\beta f(x) - 1)\pi(x) + \\ &\quad \sum_{\alpha\widetilde{h}(x) < \min(\beta f(x),1)} \left(\alpha\widetilde{h}(x) - 1\right)\pi(x). \end{aligned} \tag{8}$$

Since $\sum_x \pi(x) = 0$, the sum for the third case where 1 is the maximum enters Eq. 8 as the -1 in the two sums. This function is a piecewise linear spline in β, with knots at the locations where $\beta f(x) = \min\left(\alpha\widetilde{h}(x),1\right)$. The interior extrema (minima and maxima) of piecewise linear splines occur at their knot locations, as these are the only locations where their one-sided derivatives can change. Furthermore, $r(0) = \sum_x \pi(x) = 0$ and $r(\beta)$ is a constant for sufficiently large β. Therefore to maximize Eq. 8 we only need to consider the knots, βs that satisfy

$$\beta = \frac{\min\left(\alpha\widetilde{h}(x),1\right)}{f(x)}, \tag{9}$$

where x is from the training data. Note that by this formula all βs are positive, maintaining the regularization constraints.

Rather than calculating (Eq. 8) separately for every value of β satisfying Eq. 9, we can use the structure of the equation to efficiently update r for consecutive values of β. Let $B = \{\beta_l\}$ be the set of all βs satisfying Eq. 9 for their corresponding x_l's ($|B| \le I$), where $\beta_1 \le \beta_2 \le \cdots \le \beta_{|B|}$. Breaking up the sum into its components

$$\begin{aligned} O(\beta_l) &= \sum_{\beta f(x) \le \min\left(\alpha\widetilde{h}(x),1\right)} \pi(x), \\ P(\beta_l) &= \sum_{\beta f(x) \le \min\left(\alpha\widetilde{h}(x),1\right)} f_j(x)\pi(x), \\ Q(\beta_l) &= \sum_{\alpha\widetilde{h}(x) < \min(\beta f(x),1)} \pi(x), \\ R(\beta_l) &= \sum_{\alpha\widetilde{h}(x) < \min(\beta f(x),1)} \alpha\widetilde{h}(x)\pi(x), \end{aligned} \tag{10}$$

we can write Eq. 8 as $r(\beta_l) = \beta_l P(\beta_l) - O(\beta_l) + R(\beta_l) - Q(\beta_l)$. Now, as the βs are in order, the functions in Eq. 10 are easily updated. Let $W = \mathbf{I}\left(\alpha\widetilde{h}(x_j) < 1\right)$, then

$$\begin{aligned} O(\beta_{l+1}) &= O(\beta_l) - \pi(x_l), \\ P(\beta_{l+1}) &= P(\beta_l) - f(x_l)\pi(x_l), \\ Q(\beta_{l+1}) &= Q(\beta_l) + W\pi(x_l), \\ R(\beta_{l+1}) &= R(\beta_l) + W\alpha\widetilde{h}(x_l)\pi(x_l). \end{aligned}$$

Combining these formulas gives algorithm 2 for optimizing β.

Algorithm 2 Algorithm for optimizing β

```
Given $\alpha$, $\widetilde{h}(x) \in \widetilde{\mathcal{H}}_t$, $f(x) \in \mathcal{F}$ and the training instances.
For all $x$'s generate and sort the set of candidate βs, $B$, such that $\beta_1 \le \beta_2 \le \cdots \le \beta_{|B|}$ and $\beta_l = \frac{\min\left(\alpha\widetilde{h}(x_l),1\right)}{f(x_l)}$.
$O = \sum_{x_l} \pi(x_l)$
$P = \sum_{x_l} f(x_l)\pi(x_l)$
$Q = 0$
$R = 0$
$rbest = 0$
for $j = 1 \ldots |B|$ do
    $r = \beta_l P - O + R - Q$
    if $r > rbest$ then
        $rbest = r$
    end if
    $O = O - \pi(x_l)$,    $P = P - f(x_l)\pi(x_l)$
    if $\alpha\widetilde{h}(x_l) < 1$ then
        $Q = Q + \pi(x_l)$,    $R = R + \alpha\widetilde{h}(x_l)\pi(x_l)$
    end if
end for
```

5.2 Heuristics for Learner Selection

If at each boosting iteration we select from all combinations of $\widetilde{h}(x)$ and $f(x)$ we end up with an $O(T^2)$ algorithm, where T is the number of iterations. However, we can sacrifice accuracy for speed by only evaluating and selecting from a fixed sized pool of previously trained learners $\widetilde{h}(x)$ and features $f(x)$, where at each iteration the pool is randomly chosen from the full $\widetilde{\mathcal{H}}_t$ and $\mathcal{F}$.

To improve performance, instead of using a uniform sampling distribution we can apply a heuristic to focus the distribution on combinations with better potential. As Eq. 8 is composed of two sums, for r to be large the terms $f(x)\pi(x)$ and $\widetilde{h}(x)\pi(x)$ need to be large. We can consider $s_f = \sum_x f(x)\pi(x)$ and $s_h = \sum_x \widetilde{h}(x)\pi(x)$ as indicators of how well these components work with $\pi(x)$. Thus, we might expect larger r values to occur when these two score values are larger. Of course, we are discounting interactions, which is the reason for the combination.

Using these score values, we can order all $\widetilde{h}(x)$ and $f(x)$ separately, and sample such that learners and features with better scores are more likely to be selected. We opted for a polynomial weighting

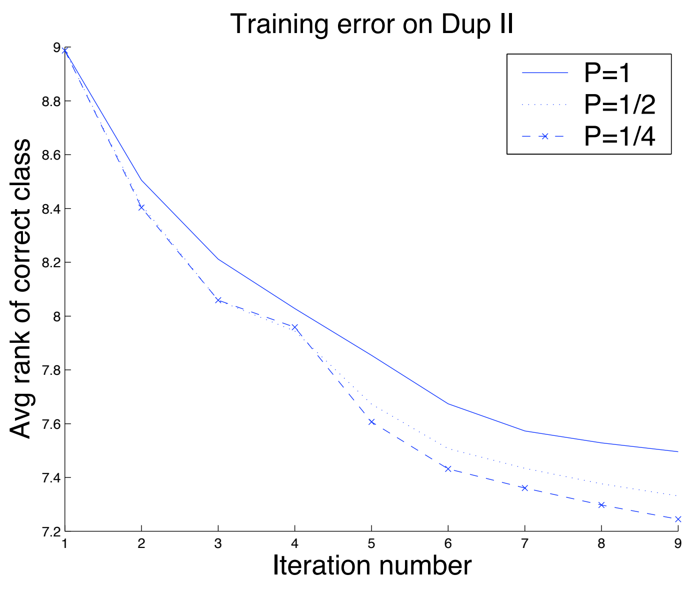


Figure 2: These plots show convergence of training error for 3 values of the heuristic selection pressure value p. These plots are averages over 10 runs and are typical of the other training data sets as well. As can be seen the heuristic improves the convergence rate of the training error.

method. Thus, for example, all $f \in \mathcal{F}$ are sorted by their score and are assigned a number based on the rank of these scores,$(maxrank - rank)/maxrank$, that gives each f an equally sized bin in the range 0 and 1. Given a random number $\xi \sim U(0,1)$, we calculate $\xi^p$and select the f that corresponds to the bin this number falls into. Here $p < 1$ can be construed as a selection pressure, where bins corresponding to higher scores are more likely to be selected. Figure 2 demonstrates the effect of different values of the p parameter in one of our experiments.

6. Face Recognition Experiments

We present experiments on the combination of face recognizers, comparing the binary learner with the MWGR learner. Given an image of a face, a face recognizer assigns a similarity score to each of the faces it is trained to recognize. These scores give a linear order or ranking to the gallery of faces. Different face recognition algorithms have different performance characteristics. Thus, we explore how combining the outputs of multiple face recognizers can improve recognition accuracy.

6.1 Algorithm Methods

We consider a data set I of face images (queries) to train on. For each query image, i in I, we need to rank all $u \in U$ faces in the gallery. In rankboost the favor function, Φ, that specifies output loss, is a function of the query and the item to be ranked. Therefore, the notational convention is to combine the query, i, and the item to be ranked, u, as an instance, $x \equiv (i,u)$. As such, $f_j(x) = f_j((i,u))$ is the

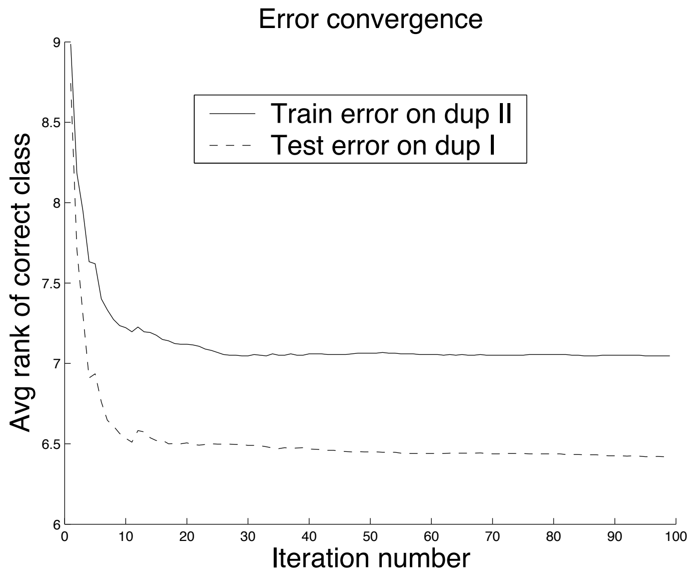


Figure 3: This plot is typical of the convergence behavior of rankboost with MWGR on the FERET data. Both training and test errors tended to converge within 10-30 iterations of boosting with no significant post-convergence divergence.

rank assigned to identity u for query image i by recognition algorithm j. As there is only one correct identity, we only care about the ranking of the one correct identity for each query. We set the favor function as stated by Freund et al. (2003) for this type of output loss. Let u^* be the the correct identity for training image i, then $\Phi\left(\left(i,u\right),\left(i,u^*\right)\right)=+1$ and $\Phi\left(\left(i,u^*\right),\left(i,u\right)\right)=-1$ for all $u\neq u^*$, setting all remaining elements of $\Phi\left(x',x''\right)=0$. That is, the correct identity of a query image is given positive favor compared to all other identities for that image, while all other rankings, including interactions between training images, are given zero favor. Note that since there is no favor interaction between queries (different i's); $\Phi\left(x',x''\right)$ is effectively a function of 3 variables, $\left(i,u',u''\right)$.

Both weak learners were trained for 100 iterations, giving them ample time to converge. See Figure 3 for an illustration of convergence times. The binary learner was trained as specified by Freund et al. (2003). At each iteration the MWGR learner was selected from a pool of candidate combinations of $f(x)$ and $\widetilde{h}(x)$, with a selection pressure of $p=0.5$. For each candidate, first β was optimized with $\alpha=1$, then α was optimized using the optimized β. The candidate with the most positive r value was always selected. This is summarized in algorithm 3.

6.2 Experimental Setup

FERET (Phillips et al., 2000) was a government sponsored program for the evaluation of face recognition algorithms. In this program commercial and academic algorithms were evaluated on their ability to differentiate between 1,196 individuals. The test consisted of different data sets of varying difficulty, for a total of 3,816 different images. The data sets, in order of perceived difficulty, are: the *fafb* data set of 1,195 images which consists of pictures taken the same day with different facial

Algorithm 3 Algorithm for selecting learner from pool.

```
Given *poolsize* and selection pressure.
Calculate $s_{f_j}$ and $s_h$ for all rank features and existing learners.
**for** $p = 1 \dots poolsize$ **do**
    Generate $d_f \sim U(0,1)$ and $d_h \sim U(0,1)$
    Select which $h = \min\left(\alpha\widetilde{h}(x),\, \beta f(x),\, 1\right)$ to try, using $s_{f_j}, s_h, u_f, u_h$.
    Setting $\alpha = 1$, optimize $\beta$ (algorithm 2)
    Keeping the optimized $\beta$, optimize $\alpha$ (algorithm 2)
    Get $r$ of learner with this $\alpha, \beta$
    **if** $r > 0$ and $r > rbest$ **then**
        $rbest = r, hbest = h, \widetilde{hbest} = \min\left(\alpha\widetilde{h}(x),\, \beta f(x)\right)$
    **end if**
**end for**
$h_t = hbest, \widetilde{\mathcal{H}}_{t+1} = \widetilde{\mathcal{H}}_t \cup \left\{\widetilde{hbest}\right\}$
```

expressions; the *fafc* data set of 194 images that contains pictures taken with different cameras and lighting conditions; the *dup I* data set of 488 images that has duplicate pictures taken within a year of the initial photo; and the most difficult, the *dup II* data set of 234 images which contains duplicate pictures taken more than a year later. Note that in our experiments we separate the images of dup II from the dup I data set, unlike the FERET study where dup II was also a subset of dup I.

The FERET study evaluated 10 baseline and proprietary face recognition algorithms. The baseline algorithms consisted of a correlation-based method and a number of eigenfaces (principle components) methods that differ in the internal metric they use. Of the proprietary algorithms, most were from different academic institutions and one was commercial. Of the 10 algorithms we selected three dominant algorithms. From the baseline algorithms we chose to use the *ANM* algorithm which uses a Mahalanobis distance variation on angular distances for eigenfaces (Moon and Phillips, 2001). While this algorithm's performance is not distinctive, within the class of baseline algorithms it was strong. Moreover, in accuracy with respect to average rank of the correct class on the dup I data set it demonstrated superior performance to all other algorithms. The other two algorithms we used were the University of Maryland's 1997 test submission (UMD) and the University of Southern California's 1997 test submission (USC). These algorithms clearly outperformed the other algorithms. UMD is based on a discriminant analysis of eigenfaces (Zhao et al., 1998), and USC is an elastic bunch graph matching approach (Wiskott et al., 1997).

The outputs of the 10 face recognizers on the four FERET data sets, fafb, fafc, dup I and dup II were the data for the experiments. Thus, we never had access to the actual classifiers, only to data on how they ranked the different faces in these data sets.

We conducted experiments based on homogeneous and heterogeneous data sets, testing the efficiency and robustness (adaptivity) of the MWGR procedure. For the homogeneous case we took all 4 FERET data sets and randomly shuffled them together. We call this the homogeneous data set as both the training and testing data are selected from the same combined pool. On this combined data set we did 4-fold cross validation. For each fold 75% of the data was used for training and the rest for testing. We combined the results of all four runs together for evaluation purposes.

For the heterogeneous case, in each experiment one of the FERET data sets was selected as a training set and another data set was selected for testing. This gave 12 experiments (not including training and testing on the same data set) per group of face recognizers, where we get combinations of training on easy data sets and testing on hard data sets, training on hard and testing on easy data sets, and training and testing on hard data sets. To reduce noise in our experiments the training ranks were truncated at 150, and outliers were removed.

In face recognition and other classification applications usually only the top ranks are important. Thus, in evaluating the results we focused on the top 30 ranks. All ranks returned by a boosted combiner for the correct class above 30 were truncated to 30.

In evaluating the performance of the combiners not all the test data are equally useful. We consider the following two cases as non informative. When the two best face recognizers, UMD and USC both give the correct class a rank of 1 there is very little reason for the combined rank to be different. Also when both the binary-learner-based combiner and the MWGR-based combiner give the correct class a rank greater than the truncation value (30) it makes little sense to compare between the combiners. The testing data was filtered to remove these cases, and the results are presented without them.

Before presenting the results, it should be said that rankboost with both learner types gives ranking functions that significantly outperform all the individual face recognition algorithms. In addition, in our tests both learners also clearly outperformed other standard rank combination methods, such as the Borda count, highest rank and logistic regression (Ho et al., 1992).

We present two sets of experiments—the combination of the 3 selected classifiers (ANM, UMD and USC) and the combination of all 10 classifiers. These are qualitatively different tasks. In combining 3, we seek to capitalize on the unique strengths of each classifier. In combining 10, we are introducing classifiers which may be correlated and classifiers which are comparably much noisier. The size of the pool was 6 when we combined 3 classifiers and 20 when combining all 10 classifiers.

For all experiments we measure the average rank of the correct class for both learners:

$$A = \frac{1}{N}\sum_{i=1}^{N} \min\left\{Rank\left(x_i^*\right), 30\right\},$$

where $Rank\left(x_i^*\right)$ is the rank of the correct class for query i, and the sum is over all useful test queries, as described above. The average rank difference between the learners is calculated to show the improvement in performance. To evaluate the significance of the improvement we ran paired one-sided t-tests and evaluated the significance of the p-value (a value less than 0.05 is significant). In addition we show the standard deviation of the rank difference.

6.3 Results

In the experiments with the homogeneous data sets, combining all classifiers gives an improvement in the average rank of the correct class of 0.296 for MWGR, with a standard deviation of 3.9, a paired t-test statistic of 1.76 and p-value of 0.039, where combining the ANM, UMD and USC classifiers gives an average rank improvement of 0.1 for MWGR, with standard deviation of 3.6, a paired t-test statistic of 0.68 and p-value of 0.24.

Table 2 contains the results for combining the 3 classifiers in the experiments with heterogeneous data sets (compare the combiner results in columns *bin mean* and *mwgr mean* with the aver-

	dup i	dup ii	fafb	fafc
ANM	9.52	18.14	10.88	19.71
ARL	16.85	16.96	8.94	28.02
EFAVG	15.02	20.61	14.43	26.17
EFML1	18.23	22.21	16.23	17.39
EFML2	15.85	20.33	12.21	19.78
EXCA	11.9	16.28	11.67	20.30
MSU	17.64	23.44	6.81	14.27
RUT	17.87	16.48	12.93	22.79
UMD	13.21	13.74	4.57	8.96
USC	12.44	6.85	5.84	5.17

Table 1: The best average rank of the correct class on the different data sets for all constituent face recognition systems.

age rank of the correct class for all constituent classifiers in Table 1). The *diff mean* column contains the improvement of MWGR over the binary learner in terms of average rank of the correct class. Of the 12 experiments, we see an improvement in 10 cases. Six of those 10 have significant p-values. The two no improvement experiments do not have significant p-values. Table 3 contains the results for combining all 10 classifiers. Of the 12 experiments, we see an improvement for MWGR in 11 cases. Eight or nine of those 11 have significant p-values. The one no improvement experiment does not have a significant p-value. It is interesting to note that we do not seem to see overfitting when increasing the number of constituents. In some case we see improvement, in others we see slight degradation, but all in all the combiner seems resilient to the noise of adding less informative constituents.

All sets of experiments, homogeneous data, heterogeneous data sets, combining 3 select recognizers and combining all 10 recognizers at once yielded significant improvements in accuracy, as is visible in the change in the average rank of the correct class and the significance of the statistical tests.

7. Information Retrieval Experiments

The annual Text REtrieval Conference (TREC) generates high-quality retrieval results of different systems on different retrieval tasks (Voorhees and Harman, 2001). We use the result data sets of the TREC-2001 web ad hoc task that uses actual web queries taken from web logs. This task has been used in other rank fusion experiments (Renda and Straccia, 2003). As did Renda and Straccia (2003) we combine the results of the following top 12 systems: iit01m, ok10wtnd1, csiro0mwal, flabxtd, UniNEn7d, fub01be2, hum01tdlx, JuruFull, kuadhoc, ricMM, jscbtawtl4, apl10wd.

Similar to other TREC information retrieval tasks, the TREC-2001 ad hoc task consists of 50 queries. For each query a list of relevant documents is supplied. Each system returns an ordered list of 1,000 documents for each query. The fusion goal is to combine these 12 individual lists into one list of 1,000 documents, which hopefully has greater precision than the individual systems. For the

test set	train set	bin mean	mwgr mean	diff mean	diff std	pval
dup i	dup ii	7.19	5.96	**1.22**	5.22	.7e-4
dup i	fafb	7.36	6.21	**1.14**	5.15	.001
dup i	fafc	8.31	6.27	**2.04**	5.51	.1e-6
dup ii	dup i	5.25	5.38	-.12	4.0	.658
dup ii	fafb	5.15	4.4	**0.74**	4.61	.018
dup ii	fafc	5.19	4.95	**0.23**	5.1	.275
fafb	dup i	2.62	2.22	**0.39**	3.09	.115
fafb	dup ii	3.38	2.60	**0.78**	3.79	.029
fafb	fafc	2.60	2.28	**0.32**	2.81	.142
fafc	dup i	2.85	2.78	**0.06**	3.33	.424
fafc	dup ii	2.40	2.43	-.03	2.27	.537
fafc	fafb	2.56	2.03	**0.52**	2.57	.028

Table 2: Results of combining the ANM, UMD, and USC classifiers using individual FERET data sets.

test set	train set	bin mean	mwgr mean	diff mean	diff std	pval
dup i	dup ii	6.73	5.89	**0.84**	4.79	.009
dup i	fafb	8.06	7.09	**0.96**	6.01	.014
dup i	fafc	6.68	4.87	**1.81**	5.24	.5e-6
dup ii	dup i	5.75	5.67	**0.08**	4.61	.408
dup ii	fafb	6.24	5.47	**0.76**	4.22	.009
dup ii	fafc	5.86	5.31	**0.55**	4.87	.074
fafb	dup i	2.67	2.07	**0.59**	2.97	.033
fafb	dup ii	3.56	2.45	**1.11**	4.51	.012
fafb	fafc	2.68	2.17	**0.51**	2.03	.01
fafc	dup i	3.36	2.51	**0.85**	3.23	.007
fafc	dup ii	3.17	2.95	**0.22**	3.95	.3
fafc	fafb	2.22	2.23	-0.01	2.08	.52

Table 3: Results of combining all 10 classifiers using individual FERET data sets.

50 queries of the TREC-2001 ad hoc task, the number of relevant documents that intersect with the union of system results range between 662 and 2664.

7.1 Methods

In the information retrieval task the favor function needs to show favor for relevant documents while disregarding other documents. Thus, we can set the favor function similarly to the way it was set in

JuruFull	UniNEn7d	apl10wd	csiro0mwa1	flabxtd	fub01be2
0.759	0.763	0.735	0.721	0.741	0.734
hum01tdlx	iit01m	jscbtawt14	kuadhoc2001	ok10wtnd1	ricMM
0.760	0.762	0.761	0.727	0.745	0.765

Table 4: Normalized mean average precision for each constituent.

the face recognition classification task. Consider a data set with I queries to train on. For each query, i in I, we need to rank all $u \in U$ documents. An instance in this case is a pair, $x \equiv (i,u)$. For each u^*a relevant document and u an irrelevant document for query i, we set $\Phi\left((i,u),(i,u^*)\right) = +1$ and $\Phi\left((i,u^*),(i,u)\right) = -1$, setting all remaining elements of $\Phi\left(x_0,x_1\right) = 0$. Thus, all relevant documents are given a positive favor with respect to irrelevant documents, while all other rankings, including interactions between relevant documents and interactions between irrelevant documents are given zero favor.

As the task has only 50 queries, rather than separating the data into train and test, we opted to do a cross validation performance evaluation. We use 5-fold cross validation on the *normalized mean average precision* measure (Salton and McGill, 1983), a standard IR measure which accounts for the precision (quantity of correct documents retrieved) and their rank position in the list. It is described by the following equation:

$$AveP = \frac{\sum_{r=1}^{N}\left(Prec(r) \times rel(r)\right)}{number\,of\,relevant\,documents}$$

where r is the rank, N is the number of documents retrieved, $rel()$ is a binary function of the relevance of the document at rank r,and $Prec$ is the precision ([number of relevant documents retrieved] / [number of documents retrieved]) at a given cut-off rank. It is clear that higher values of $AveP$ are better. Note that for purposes of normalization we assigned unretrieved relevant documents a constant rank.

As the binary and MWGR weak learners had significantly different convergence properties on this task (see Figure 4) MWGR was trained for 100 iterations and the binary learner for 300 iterations. As in the FERET experiments, MWGR was optimized using algorithm 3, with a selection pressure of $p = 0.5$.

7.2 Results

As seen in Figure 4, the MWGR learner converges in significantly less iterations than the binary learner. This could possibly be attributed to the fact that the MWGR is a more complex function that can incorporate the rankings of multiple classifiers in each learner. Also the MWGR function is not tuned to particular rank cutoffs, whereas the binary learner is, so the MWGR can better accommodate the variety in the 1000 ranks being considered.

The normalized mean average precision for the MWGR after 100 iterations was 0.8537 and it was 0.8508 for the binary learner after 300 iterations. Compare these results with the precision of the constituents in Table 4. Both weak learners had a performance rate of change of approximately $3 * 10^{-5}$ on their final iteration (better for MWGR). A paired t-test on the cross validation results of the two learners gives a statistically significant p-value of 0.007 in favor of MWGR.

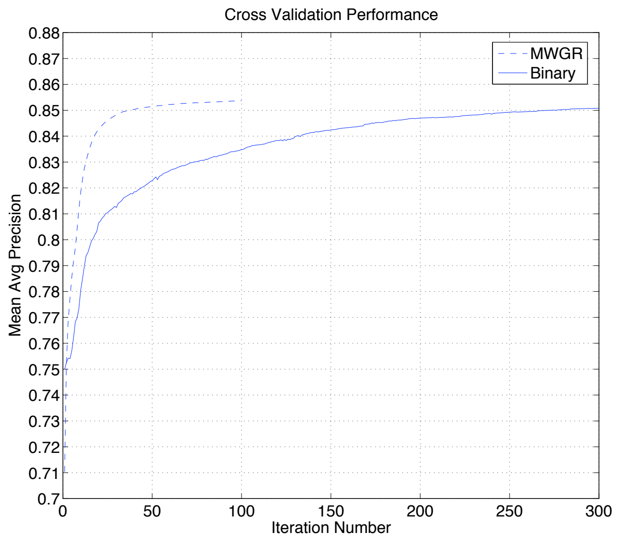


Figure 4: The cross validation mean average precision score of the two weak learners, MWGR and binary, as a function of boosting iteration number.

8. Discussion

The question of how to combine ordinal data has become an active focus of research in machine learning, as applications in pattern recognition, information retrieval and other domains have come to the forefront. A particular question of importance is how can the structure of ranks be correctly exploited to maximize performance. The semi parametric nature of rankboost offers the possibility to generate arbitrarily flexible ranking functions. But as observed this flexibility comes at a cost of significant overfitting without further regularization. Freund et al. (2003) demonstrate that successful generalization only occurs when the resulting ranking functions are constrained to be monotonic. This constraint can be thought of as a regularization that incorporates prior knowledge on the interpretation of ranks and as such, how they can be combined.

We present a regularization framework based on the concept of consistency in confidence and preference. Ranking functions with this property show a consistency in how they treat the preference and relative confidence exhibited by constituents. We prove that under a natural interpretation of preference and confidence for ranks, this consistency property of the combiner is equivalent to monotonicity and concavity of its ranking function.

We enhance rankboost by designing a weak ranking learner that exhibits consistency in preference and confidence. A computational advantage of this weak learner, called minimum weighted group ranks (MWGR) is that its parameters can be individually optimized readily with respect to the rankboost criteria, allowing it to be tested on real-world data.

In our first experiments we compare the original rankboost binary weak learner with MWGR on a task of combining the output of multiple face recognition algorithms from the FERET study. We conducted experiments on homogeneous data, testing the intrinsic efficiency of the MWGR procedure. We also conducted experiments on heterogeneous data, testing the robustness or adaptivity of the procedure. In almost all cases we see that MWGR shows improved performance compared with the binary weak learner, whether combining three or all of the face recognizers, confirming the utility of this monotonic and concave learner. Our second experiment was on an Information Retrieval task taken from the TREC conference. In this task we see MWGR converges in significantly less iterations and generates statistically significant improved performance.

Final Words

Ofer Melnik and Cun-Hui Zhang are very saddened that our colleague and friend Yehuda Vardi passed away before he could give this paper his final stamp of approval. He was very enthusiastic about this research and would have been very pleased to see it come to fruition.

Acknowledgments

This research is partially supported by NSF Grants DMS 04-05202, DMS 05-04387 and, DMS 06-04571, ONR Grant N00014-02-1-056 and NSA Grant H98230-04-1-0041. Ofer Melnik would also like to thank DIMACS for their support in this research. The authors are also grateful to the editor and reviewers for their constructive suggestions which improved the presentation of the results.

Appendix A.

In this paper we showed how three-point consistency in preference and confidence implies concave and monotonic ranking functions. For two decision problems involving two pairs of rank vectors, the four-point consistency property implies the following constraints for a ranking function

$$\left\{\begin{array}{l} G(\mathbf{y}) < G(\mathbf{y}') \\ \mathbf{y}'_{\mathbf{B}} - \mathbf{y}_{\mathbf{B}} \le 0 < \mathbf{y}'_{\mathbf{A}} - \mathbf{y}_{\mathbf{A}} \\ \mathbf{z}_{\mathbf{A}} \le \mathbf{y}_{\mathbf{A}}, \mathbf{y}'_{\mathbf{A}} - \mathbf{y}_{\mathbf{A}} \le \mathbf{z}'_{\mathbf{A}} - \mathbf{z}_{\mathbf{A}} \\ \mathbf{y}_{\mathbf{B}} \le \mathbf{z}_{\mathbf{B}}, \mathbf{z}_{\mathbf{B}} - \mathbf{z}'_{\mathbf{B}} \le \mathbf{y}_{\mathbf{B}} - \mathbf{y}'_{\mathbf{B}} \end{array}\right. \Rightarrow G(\mathbf{z}') - G(\mathbf{z}) > G(\mathbf{y}') - G(\mathbf{y}) \qquad (11)$$

where $\mathbf{y}$ and $\mathbf{y}'$ are the rank vectors from the first decision problem and $\mathbf{z}$ and $\mathbf{z}'$ are the rank vectors from the second decision problem.

Unlike the three-point case, the four-point consistency property does not imply a clearly recognizable functional form for the ranking function. What we can say about it though is that as the constraints are linear, in the same way that concavity and monotonicity in the weak learner conferred the same properties to the ranking function, a weak learner that satisfies Eq. 11 will also confer those properties to the ranking function that uses it with positive weights.

References

K. Arrow. *Social Choice and Individual Values*. Wiley, 1951.

C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proc. 10th Intl. World Wide Web Conf.*, pages 613–622, 2001.

Y. Freund, R. Iyer, R.E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.

Y. Freund and R.E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 148–156, 1996.

T. K. Ho. *A Theory of Multiple Classifier Systems and Its Application to Visual Word Recognition*. PhD thesis, State University of New York at Buffalo, May 1992.

T. K. Ho, J. J. Hull, and S. N. Srihari. Combination of decisions by multiple classifiers. In H. S. Baird, H. Bunke, and K. Yamamoto (Eds.), editors, *Structured Document Image Analysis*, pages 188–202. Springer-Verlag, Heidelberg, 1992.

J. Kittler and F. Roli, editors. *Multiple Classifier Systems, Lecture Notes in Computer Science 1857*, 2000. Springer.

R. Meir and G. Ratsch. *Advanced Lectures in Machine Learning, Lecture Notes in Computer Science 2600*, chapter An introduction to boosting and leveraging, pages 119–184. Springer, 2003.

O. Melnik, Y. Vardi, and C-H. Zhang. Mixed group ranks: Preference and confidence in classifier combination. *IEEE Pattern Analysis and Machine Intelligence*, 26(8):973–981, 2004.

H. Moon and P.J. Phillips. Computational and performance aspects of PCA-based face-recognition algorithms. *Perception*, 30:303–321, 2001.

P.J. Phillips, H. Moon, S.A. Rizvi, and P.J. Rauss. The FERET evaluation methodology for face-recognition algorithms. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:1090–1104, 2000.

M.E. Renda and U. Straccia. Web metasearch: Rank vs. score based rank aggregation methods. In *18th Annual ACM Symposium on Applied Computing (SAC-03)*, pages 841–846, Melbourne, Florida, USA, 2003. ACM.

G. Salton and J.M. McGill. *Introduction to Modern Information Retrieval*. Addison Wesley Publ. Co., 1983.

E.M. Voorhees and D.K. Harman, editors. *NIST Special Publication 500-250: The Tenth Text REtrieval Conference (TREC 2001)*, number SN003-003-03750-8, 2001. Department of Commerce, National Institute of Standards and Technology, Government Printing Office. URL `http://trec.nist.gov`.

L. Wiskott, J.-M. Fellous, N. Kruger, and C. von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(7):775–779, 1997.

W. Zhao, A. Krishnaswamy, R. Chellappa, D. Swets, and J. Weng. *Face Recognition: From Theory to Applications*, chapter Discriminant Analysis of Principal Components, pages 73–86. Springer-Verlag, Berlin, 1998.

Journal of Machine Learning Research 8 (2007) 813-839 Submitted 5/06; Revised 2/07; Published 4/07

Gini Support Vector Machine: Quadratic Entropy Based Robust Multi-Class Probability Regression

Shantanu Chakrabartty SHANTANU@MSU.EDU
Department of Electrical and Computer Engineering
Michigan State University
East Lansing, MI 48824, USA

Gert Cauwenberghs GERT@UCSD.EDU
Division of Biological Sciences
University of California San Diego
La Jolla, CA 92093-0357, USA

Editor: Alex Smola

Abstract

Many classification tasks require estimation of output class probabilities for use as confidence scores or for inference integrated with other models. Probability estimates derived from large margin classifiers such as support vector machines (SVMs) are often unreliable. We extend SVM large margin classification to *Gini*SVM maximum entropy multi-class probability regression. *Gini*SVM combines a quadratic (Gini-Simpson) entropy based agnostic model with a kernel based similarity model. A form of Huber loss in the *Gini*SVM primal formulation elucidates a connection to robust estimation, further corroborated by the impulsive noise filtering property of the reverse water-filling procedure to arrive at normalized classification margins. The *Gini*SVM normalized classification margins directly provide estimates of class conditional probabilities, approximating kernel logistic regression (KLR) but at reduced computational cost. As with other SVMs, *Gini*SVM produces a sparse kernel expansion and is trained by solving a quadratic program under linear constraints. *Gini*SVM training is efficiently implemented by sequential minimum optimization or by growth transformation on probability functions. Results on synthetic and benchmark data, including speaker verification and face detection data, show improved classification performance and increased tolerance to imprecision over soft-margin SVM and KLR.

Keywords: support vector machines, large margin classifiers, kernel regression, probabilistic models, quadratic entropy, Gini index, growth transformation

1. Introduction

Support vector machines (SVMs) have gained much popularity in the machine learning community as versatile tools for classification and regression from sparse data (Boser et al., 1992; Vapnik, 1995; Burges, 1998; Schölkopf et al., 1998). The foundations of SVMs are rooted in statistical learning theory (Vapnik, 1995) with also connections to regularization theory (Girosi et al., 1995; Pontil and Verri, 1998a). The principle of structural risk minimization provides bounds on generalization performance which make SVMs well suited for applications with sparse training data (Joachims, 1997; Oren et al., 1997; Pontil and Verri, 1998b).

Several classification problems in machine learning require estimation of multi-class output probabilities. Besides their use as confidence scores in classification, the class probability estimates can also be used in combination with other probabilistic models such as hidden Markov models for inference across graphs. For instance text-independent speaker verification systems require normalized classifier scores to be integrated over several speech frames in an utterance (Auckenthaler et al., 2000) to arrive at global acceptance/rejection scores. Even though SVMs have been successfully applied for the task of speaker verification (Schmidt and Gish, 1996; Gu and Thomas, 2001), the cumulative scores generated by SVMs are susceptible to corruption by impulse noise, which increases false acceptance rate.

Multi-class extensions to SVM classification have been formulated, based on 'one vs. all' (Weston and Watkins, 1998; Crammer and Singer, 2000) or 'one vs. one' (Schölkopf et al., 1998; Dietterich and Bakiri, 1995; Allewin et al., 2000; Hsu and Lin, 2002) methods. In its general setting multi-class SVMs generate unnormalized and biased estimates of class conditional probabilities (Platt, 1999a). Calibration and moderation methods have been proposed to arrive at class probability estimates from the trained SVM classifier (Kwok, 1999; Platt, 1999a). For instance Platt (1999a) applied sigmoidal regression to the output of an SVM and showed a performance comparable to regularized maximum likelihood kernel methods (Jaakkola and Haussler, 1999; Zhu and Hastie, 2002). Vapnik has proposed a probability regression technique based on mixture of cosine functions (Vapnik, 1995), where the coefficients of the cosine expansion minimize a regularized function. A drawback of these methods is their difficulty in embedding other inference models like graphical models where re-estimation of SVM parameters can be naturally performed. Kernel logistic regression (KLR) (Jaakkola and Haussler, 1999) provides such a framework to estimate probabilities and can be easily embedded into graphical models with its parameters estimated using an expectation-maximization (EM) like procedure (Jordan and Jacobs, 1994). However, one of the disadvantages of KLR is that the kernel expansion is non-sparse in the data making regression infeasible for large classification problems. A Bayesian learning framework using relevance determination on linear models more general than kernel regression (Tipping, 2001) produces a very sparse expansion but involves significant computation during training that does not scale well to very large data. Recently sparse Gaussian process based methods have been reported (Lawrence et al., 2003), that alleviate scalability problems of relevance determination through use of greedy optimization techniques.

The purpose of this paper is to describe a unifying framework for SVM based classification that directly produces probability scores. Previous work in this area used Shannon entropy in a large margin framework (Jebara, 2001) which led directly to KLR and hence inherited its potential disadvantages of non-sparsity. One of the important contributions of the paper is exploration of links between maximum entropy based learning techniques and large margin classifiers with extensions to quadratic based impurity functions. Within this framework the paper introduces the *Gini* Support Vector Machine (*Gini*SVM) (Chakrabartty and Cauwenberghs, 2002), a large margin classifier based on a quadratic entropy formulation combined with kernel based quadratic distance. At the core of *Gini*SVM is a margin normalization procedure that moderates the output of the classifier. Training *Gini*SVM entails solving a quadratic programming problem analogous to soft-margin SVM. We also present algorithms for training *Gini*SVM classifiers with multiplicative updates, and by growth transformation on polynomial objective functions.

The paper is organized as follows: Section 2 introduces a supervised discriminative framework for obtaining classifiers that produce conditional probability scores. Section 4 introduces *Gini*SVM and derives its normalization properties based on a reverse water-filling algorithm. Section 5 presents algorithms for *Gini*SVM training based on conventional sequential minimum optimization (SMO) and a novel multiplicative update algorithm. Section 6 compares the performance of *Gini*SVM for benchmark UCI databases, a face detection and a text-independent speaker verification task. Section 7 provides concluding remarks with future directions.

2. Generalized Maximum Entropy Based Supervised Learning

In the framework of supervised learning, the learner is provided with a training set of feature vectors $\mathcal{T} \subset \mathcal{X} : \mathcal{T} = \{\mathbf{x}_i\}, i = 1,..,N$ drawn independently from a fixed distribution $P(\mathbf{x}), \mathbf{x} \in \mathcal{X}$. The formulation presented here assumes a countable set $\mathcal{X}$ even though it generalizes to uncountable sets. Also provided to the learner is a set of soft (or possibly hard) labels that represent conditional probability measures $y_{ik} = P(C_k|\mathbf{x}_i)$ defined over a discrete set of classes $C_k, k = 1,..,M$. The labels therefore are normalized and satisfy $\sum_{k=1}^{M} y_{ik} = 1$. The aim of the learner is to choose a finite set of regression functions $P = \{P_k(\mathbf{x})\}, k = 1,..,M$ that accurately predict the true conditional probabilities $P(C_k|\mathbf{x})$. For this purpose the learner uses a distance metric $D_Q : R^M \times R^M \rightarrow R$ that embeds prior knowledge about the topology of the feature space. Since the prior labels y_{ik} are available only for the training set, the learner also defines an agnostic (non-informative) distance metric $D_I : R^M \times R^M \rightarrow R$ which does not assume any knowledge of the training set. The embedded agnostic prior is consistent with maximum entropy principles (Jaynes, 1957; Pietra and Pietra, 1993) and enforces smoothness constraints on the the function $P_k(\mathbf{x})$ by avoiding solutions that over-fit to the training set. Estimating the probability functions $P = \{P_k(\mathbf{x})\}$ entails a training procedure involving minimization of a joint distance metric and is given by

$$\min_P G(P) = \min_P [D_Q(Y,P) + \gamma D_I(P,U)]. \tag{1}$$

Here $Y : R^{|\mathcal{T}|} \times R^M$ is a matrix of prior labels $y_{ik},\ i = 1,..,N,\ k = 1,..,M$, and U denotes a uniform distribution given by $U_k(\mathbf{x}) = 1/M,\ \forall k = 1,..,M$. $\gamma > 0$ is a hyper-parameter that determines a trade-off between the prior and agnostic distance metrics. Minimizing the cost function (1) leads to a solution P that is not only close to a prior distribution with respect to the distance metric $D_Q(.,.)$ but is also close to the non-informative (agnostic) uniform distribution U. In addition, the maximum entropy framework (Pietra and Pietra, 1993) allows to impose linear constraints on the optimization problem (1) based on cumulative statistics defined on the training set. One linear constraint equates the frequency of occurrence of a class $k = 1,..,M$ under the distribution P to an equivalent measure under the prior distribution y_{ik}. This first constraint can be written to express equivalence between average estimated probabilities and empirical frequencies for each class over the training set

$$\sum_{i=1}^{N} P_k(\mathbf{x}_i) = \sum_{i=1}^{N} y_{ik}, \quad k = 1,\dots M \tag{2}$$

under the assumption that all features $\mathbf{x} \in \mathcal{X}$ are equally likely. A second set of linear constraints expresses boundary and normalization conditions for valid probability distributions

$$P_k(\mathbf{x}) \geq 0, \quad k = 1,\dots M, \tag{3}$$

$$\sum_{k=1}^{M} P_k(\mathbf{x}_i) = 1 \tag{4}$$

where the additional inequality constraint $P_k(\mathbf{x}) \leq 1,\ k = 1,\dots M$ is subsumed by the normalizing equality constraint.

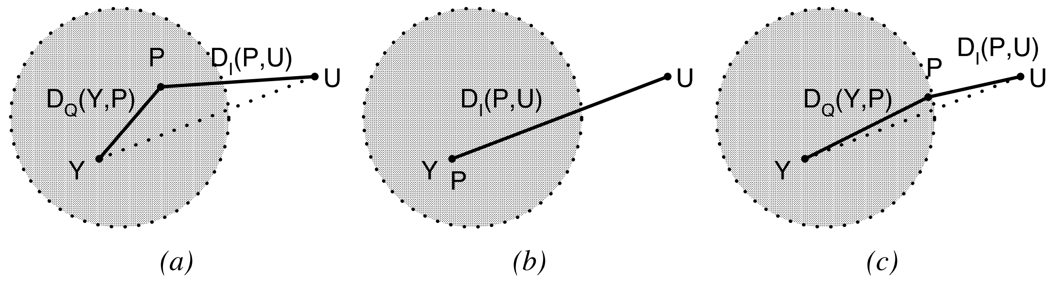


Figure 1: Generalized framework for maximum entropy probability regression. *(a)*: Solution P lies in the constraint space shown as a sphere such that the total distance to the distribution Y and U is minimized. *(b)*: Solution for $\gamma = 0$, where P coincides with Y. *(c)*: Solution for $\gamma \to \infty$, projecting U onto the constraint space.

Pictorially the solution to the optimization problem (1) is shown in Figure 1. For illustration purposes the linear constraints (2), (3) and (4) are represented by the shaded circle. The distance $D_Q(Y,P)$ determines the proximity of distribution P to a prior empirical distribution Y. $D_I(P,U)$ is a distance that defines an agnostic model when any prior knowledge about prior distribution is absent. This framework is similar to the maximum entropy approach (Jaynes, 1957; Pietra and Pietra, 1993; Jebara, 2001). The possible solutions to minimizing the cost function (1) with constraints (2)-(4) are shown in Figure 1 where the solution P lies within or at the boundary of the constraint space. Note that the constraint space also includes the prior distribution Y. Under non-degenerate conditions the agnostic U distribution will lie outside the constraint space. The value of the hyper-parameter $\gamma > 0$ influences the location of the solution P with respect to the prior Y and agnostic U distributions. As we will see further below, the parameters also determine the sparsity and generalization performance of classifiers defined by parameters P. As shown in Figure 1, for $\gamma = 0$, the solution is the prior distribution Y and thus over-fits to the training set. For the case when $\gamma \to \infty$, the solution is equivalent to maximum entropy, which is the projection of U on the constraint space.

The solution to the optimization (1) is obtained by first order Karush-Kuhn-Tucker (KKT) conditions (Bertsekas, 1995) with respect to the probability functions $P = \{P_k(\mathbf{x})\}$ and is given by

$$\gamma \frac{\partial D_I(P,U)}{\partial P_k(\mathbf{x})} = -\frac{\partial D_Q(Y,P)}{\partial P_k(\mathbf{x})} + b_k - z(\mathbf{x}) + \beta_k(\mathbf{x}). \tag{5}$$

Here b_k represent Lagrange multipliers corresponding to frequency constraints (2), $\beta_k(\mathbf{x}) \geq 0$ are Lagrange multipliers for the inequality constraints (3), and Lagrange multipliers $z(\mathbf{x})$ correspond to the normalization constraint (4). For the sake of simplicity we will assume a form of $D_I(P,U)$ that can be decomposed into independent, identically distributed (i.i.d.) components as

$$D_I(P,U) = \sum_{k=1}^{M} \sum_{\mathbf{x} \in \mathcal{T}} \Psi(P_k(\mathbf{x}), U_k(\mathbf{x})), \tag{6}$$

where $\Psi : R \times R \to R$ is a concave function. The first order condition (5) can be written as a Legendre transform (Rockefeller, 1970) with respect to $\Psi(.)$ as

$$P_k(\mathbf{x}) = \nabla \Psi^{-1} \left(\frac{1}{\gamma} \left[-\frac{\partial D_Q(Y,P)}{\partial P_k(\mathbf{x})} + b_k - z(\mathbf{x}) + \beta_k(\mathbf{x}) \right] \right). \tag{7}$$

where $\nabla\Psi^{-1}(.)$ denotes the Legendre transform with respect to $P_k(\mathbf{x})$ for the bivariate function $\Psi(.)$. The Legendre transformation is commonly used in the dual formulation of support vector machines and other kernel machines (Vapnik, 1995; Schölkopf and Smola, 2001), and we refer to $\nabla\Psi^{-1}(.)$ as the dual potential function. Note that $\nabla\Psi^{-1}(.)$ is monotonic due to the concavity of $\Psi(.)$.

Several choices exist for the prior distance metric $D_Q(.,.)$. A popular metric is a quadratic distance extensively used in kernel methods (Schölkopf et al., 1998) and as covariance functions in Bayesian methods (Jordan and Jacobs, 1994). In its general form the quadratic distance between two conditional distributions $\hat{P} = \{\hat{P}_k(\mathbf{x})\}$ and $P = \{P_k(\mathbf{x})\}$ is given by

$$D_Q(\hat{P},P) = \frac{C}{2}\sum_{k=1}^{M}\sum_{\mathbf{x},\mathbf{v}\in\mathcal{T}} K(\mathbf{x},\mathbf{v})\left[\hat{P}_k(\mathbf{x}) - P_k(\mathbf{x})\right]\left[\hat{P}_k(\mathbf{v}) - P_k(\mathbf{v})\right]. \tag{8}$$

Here $K : R^M \times R^M \to R$ represents a symmetric, positive definite kernel satisfying the *Mercer's criterion*,[1] such as a Gaussian radial basis function or a polynomial spline (Schölkopf et al., 1998; Wahba, 1998). The distance $D_Q(.,.)$ embeds prior knowledge induced by the kernel $K(\mathbf{x},\mathbf{v})$ and therefore quantifies a topology of a metric space for points $\mathbf{x},\mathbf{v} \in \mathcal{X}$.

For the quadratic form $D_Q(.,.)$ given by Equation (8) the first order conditions (7) can be written as

$$P_k(\mathbf{x}) = \nabla\Psi^{-1}\left(\frac{1}{\gamma}[f_k(\mathbf{x}) - z(\mathbf{x}) + \beta_k(\mathbf{x})]\right) \tag{9}$$

where

$$f_k(\mathbf{x}) = \sum_{i=1}^{N}\lambda_k^i K(\mathbf{x}_i,\mathbf{x}) + b_k$$

with inference parameters

$$\lambda_k^i = C[y_{ik} - P_k(\mathbf{x}_i)].$$

The Lagrange parameter function $\beta_k(\mathbf{x})$ in Equation (9) needs to ensure that the probability scores $P_k(\mathbf{x}) \geq 0 \quad \forall\mathbf{x} \in \mathcal{X}$ according to (3), and the Lagrange parameter function $z(\mathbf{x})$ needs to ensure normalized probabilities $\sum_{k=1}^{M} P_k(\mathbf{x}) = 1$ according to (4).

The set of inference parameters $\Lambda = \{\lambda_k^i\}, i = 1,..,N, \quad k = 1,..,M$ is obtained by solving (1) over the training set $\mathcal{T}$. Expressing the general form (6) for the agnostic distance $D_I(P,U)$ and the quadratic distance (8) for the prior distance $D_Q(Y,P)$ in terms of the inference parameters λ_k^i in the cost function (1) leads to a dual formulation H_d

$$H_d = \sum_{k=1}^{M}\left[\frac{1}{2C}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_k^i Q_{ij}\lambda_k^j + \gamma\sum_{i=1}^{N}\Psi(y_{ik} - \lambda_k^i/C)\right] \tag{10}$$

where $Q_{ij} = K(\mathbf{x}_i,\mathbf{x}_j)$ denote elements of the *kernel matrix* $\mathbf{Q}$. Like the primal (1), minimization of the dual H_d is subject to linear constraints (2)-(4) rewritten in terms of the inference parameters as

$$\begin{aligned} \sum_{k=1}^{M}\lambda_k^i &= 0, \quad i = 1,\ldots N, \\ \sum_{i=1}^{N}\lambda_k^i &= 0, \quad k = 1,\ldots M, \\ \lambda_k^i &\leq Cy_{ik}. \end{aligned} \tag{11}$$

1. $K(\mathbf{x},\mathbf{v}) = \Phi(\mathbf{x})\cdot\Phi(\mathbf{v})$. The map $\Phi(\cdot)$ need not be computed explicitly, as it only appears in inner-product form.

3. Kernel Logistic Regression

For a frame of reference in the comparison between different formulations of cost functions, the optimization framework given by the dual (10) subject to constraints (11) is first applied to kernel logistic regression (KLR) (Jaakkola and Haussler, 1999), with agnostic distance

$$\begin{aligned} D_I(P,U) &= \sum_{k=1}^{M} \sum_{\mathbf{x}\in\mathcal{T}} P_k(\mathbf{x}) \log \frac{P_k(\mathbf{x})}{U_{ik}} \\ &= \sum_{k=1}^{M} \sum_{\mathbf{x}\in\mathcal{T}} P_k(\mathbf{x}) \log P_k(\mathbf{x}) + \mathrm{cst} \qquad (12) \end{aligned}$$

derived from the Kullback-Leibler (KL) divergence $\Psi_{KL}(P,U) = P\log(P/U)$ for a uniform agnostic distribution $U_{ik} \equiv 1/M$. The constant term $\mathrm{cst} = N\log M$ in (12) drops in the minimization and is subsequently ignored. The probability function according to (9) is then given by

$$P_k(\mathbf{x}) = \exp\left(\frac{1}{\gamma}[f_k(\mathbf{x}) - z(\mathbf{x}) + \beta_k(\mathbf{x})]\right). \qquad (13)$$

By property of $\exp(.)$, $P_k(\mathbf{x}) \geq 0 \quad \forall \mathbf{x} \in \mathcal{X}$, and so the Lagrange multiplier $\beta_k(\mathbf{x})$ in (13) is arbitrary and can be eliminated, $\beta_k(\mathbf{x}) \equiv 0$. The other Lagrange multiplier $z(\mathbf{x})$ in (13) is determined by expressing the normalization condition $\sum_{k=1}^{M} P_k(\mathbf{x}) = 1$ which leads to a logistic model

$$P_k(\mathbf{x}) = \exp\left(\frac{1}{\gamma} f_k(\mathbf{x}))/\sum_{p=1}^{M} \exp(\frac{1}{\gamma} f_p(\mathbf{x})\right). \qquad (14)$$

Substituting the KL distance $\Psi_{KL}(.,.)$ for $\Psi(.,.)$ in the general form (10) directly leads to the dual cost function

$$H_e = \sum_{k=1}^{M}\left[\frac{1}{2C}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_k^i Q_{ij} \lambda_k^j + \gamma \sum_{i=1}^{N} (y_{ik} - \lambda_k^i/C) \log(y_{ik} - \lambda_k^i/C)\right] \qquad (15)$$

subject to the dual constraints (11).

3.1 KLR Primal Reformulation

The dual (15) derived from the general maximum entropy form (1) is identical to the dual formulation of another, closely related primal cost function for kernel logistic regression as formulated in Jaakkola and Haussler (1999). The purpose of this section is to establish the equivalence with a connection to large margin kernel machines and their interpretation in feature space (Schölkopf and Smola, 2001).

Expressing the kernel function $K(\mathbf{x},\mathbf{v}) = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{v})$ as an inner-product in a transformed feature space $\Phi(.)$, the decision functions $f_k(\mathbf{x})$ are linked to a set of M hyperplanes

$$\begin{aligned} f_k(\mathbf{x}) &= \sum_{i=1}^{N} \lambda_k^i \, K(\mathbf{x}_i, \mathbf{x}) + b_k \qquad (16) \\ &= \sum_{i=1}^{N} \lambda_k^i \, \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}) + b_k \qquad (17) \\ &= \mathbf{w}_k \cdot \Phi(\mathbf{x}) + b_k \end{aligned}$$

where $\mathbf{w}_k = \sum_{i=1}^{N} \lambda_k^i \Phi(\mathbf{x}_i)$ represent the parameters of the hyperplanes. The following proposition links the kernel logistic regression dual (15) with its equivalent primal formulation (Jaakkola and Haussler, 1999).

Proposition I: The kernel logistic regression objective function (15) is the dual derived from a primal objective function with regularized loss function

$$\begin{aligned} L_e &= \sum_{k=1}^{M} \frac{1}{2}|\mathbf{w}_k|^2 + C\sum_{i=1}^{N}\sum_{k=1}^{M} y_{ik} \log \frac{y_{ik}}{P_k(\mathbf{x}_i)} \\ &= \sum_{k=1} \frac{1}{2}|\mathbf{w}_k|^2 - C\sum_{i=1}^{N}[\sum_{k=1}^{M} y_{ik} f_k(\mathbf{x}_i) - \log(e^{f_1(\mathbf{x}_i)} + \ldots + e^{f_M(\mathbf{x}_i)})] \,. \end{aligned} \tag{18}$$

where additional constant terms in (18) have been ignored and a unity value for γ has been assumed in the probability model (14). The proof of the proposition is provided in Appendix A.

The primal uses the Kullback-Leibler (KL) divergence $\Psi_{KL}(P,U) = P\log(P/U)$ between distributions y_{ik} and $P_k(\mathbf{x}_i)$ as loss function in the regularized form (18) (Wahba, 1998; Zhu and Hastie, 2002). One of the disadvantages of the kernel logistic dual is that the KL divergence distance metric strongly penalizes solutions far away from the agnostic distribution U, leading to a non-sparse kernel expansion. A sparser kernel expansion is obtained in soft-margin support vector machines for classification. A *Gini* form of entropy as agnostic distance metric provides the connection between support vector machines and probability regression, studied next.

4. *Gini*SVM and Margin Normalization

Instead of KL divergence, a natural choice for an agnostic distance metric D_I is a quadratic form of entropy similar to the quadratic form of the prior distance metric D_Q. A *Gini* quadratic form of entropy, or impurity function, has been used extensively in natural language processing for growing decision trees (Breiman et al., 1984). The *Gini* quadratic entropy forms the basis of the *Gini*-support vector machine (*Gini*SVM) for probability regression (Chakrabartty and Cauwenberghs, 2002).

The *Gini* quadratic form of entropy $\Psi_{Gini}(P,U) = \frac{1}{2}(P-U)^2$ with uniform agnostic distribution $U_{ik} \equiv 1/M$ leads to an agnostic distance metric

$$\begin{aligned} D_I(P,U) &= \frac{1}{2}\sum_{k=1}^{M}\sum_{\mathbf{x}\in\mathcal{T}}(P_k(\mathbf{x}) - U_{ik})^2 \\ &= \frac{1}{2}\sum_{k=1}^{M}\sum_{\mathbf{x}\in\mathcal{T}} P_k(\mathbf{x})^2 + \text{cst} \end{aligned}$$

where the constant term cst $= -N/2M$ drops in the minimization. Substituting the Gini distance $\Psi_{Gini}(.,.)$ for $\Psi(.,.)$ in the general form (10) leads to the dual *Gini*SVM cost function

$$H_g = \sum_{k=1}^{M}\left[\frac{1}{2C}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_k^i Q_{ij}\lambda_k^j + \frac{\gamma}{2}\sum_{i=1}^{N}(y_{ik} - \lambda_k^i/C)^2\right] \tag{19}$$

under constraints (11).

In contrast to the KL distance in the KLR dual (15), the quadratic distance in the *Gini*SVM dual (19) allows sparse kernel expansions, where several of the inference parameters λ_k^i are driven

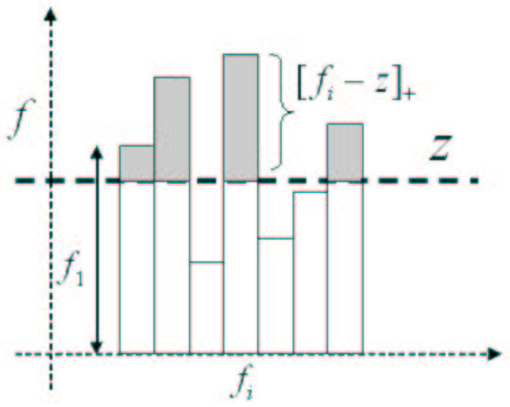


Figure 2: Illustration of reverse water-filling procedure. The level z is adjusted as to maintain the net f_i in excess of z (shaded area) at γ.

to zero by the inequality constraints in (11) corresponding to a majority of labels for which $y_{ik} = 0$. Even sparser kernel expansions could be obtained with soft-margin support vector machines for classification, owing to its slightly different quadratic cost function under linear constraints which further favors sparsity. More significantly, *Gini*SVM produces conditional probability estimates that are based on maximum entropy (1). The probability estimates themselves could be sparse, with $P_k(\mathbf{x}) = 0$ for a number of classes k depending on γ and $\mathbf{x}$, as we show next.

4.1 Margin Normalization and Reverse Waterfilling

The quadratic entropy form of $\Psi_{Gini}(.,.)$ leads to a linear Legendre transform $\nabla\Psi^{-1}(.)$ and thus a linear, rather than exponential, form of the conditional probability estimates $P_k(\mathbf{x}) = 1/\gamma/[f_k(\mathbf{x}) - z(\mathbf{x}) + \beta_k(\mathbf{x})], k = 1,..,M$. To ensure positive probabilities according to constraints (3), the Lagrange parameters $\beta_k(\mathbf{x})$ produce rectified linear probability estimates

$$P_k(\mathbf{x}) = \frac{1}{\gamma}[f_k(\mathbf{x}) - z(\mathbf{x})]_+ \tag{20}$$

where $[x]_+ = \max(x,0)$ denotes a hinge function. The remaining Lagrange parameter $z(\mathbf{x})$ is determined through a subtractive normalization procedure which solves for the normalization constraint (4)

$$\sum_{k=1}^{M} [f_k(\mathbf{x}) - z(\mathbf{x})]_+ = \gamma. \tag{21}$$

The conditions (20) and (21) are jointly satisfied by applying a reverse water-filling algorithm commonly found in communication systems (Cover and Thomas, 1991), listed in Algorithm 1 and illustrated in Figure 2. The algorithm recursively computes the normalization factor $z(\mathbf{x})$ such that the net balance of class confidence levels $f_k(\mathbf{x})$ in excess of $z(\mathbf{x})$ equals γ.

We refer to the procedure solving for $z(\mathbf{x})$ given confidence scores $f_k(\mathbf{x})$ in (21) as *margin normalization*, because of similarities between the normalization parameter γ and the margin of multi-class soft-margin support vector machines (Weston and Watkins, 1998). Unlike the *divisive*

Algorithm 1 Reverse water-filling procedure to compute normalization parameter z

```
Require: Set of confidence values $\{f_k(x)\}, k = 1,..,M$.
Ensure: $z = 0, N = 1, T = 0$
  $a = max\{f_k(x)\}$
  $\{s\} \leftarrow \{f_k(x)\} - \{a\}$
  while $T < \gamma$ & $N < M$ do
    $b = max\{s\}$
    $T \leftarrow T + N(a-b)$
    $a \leftarrow b$
    $\{s\} \leftarrow \{s\} - \{b\}$
    $N \leftarrow N + 1$
  end while
  $z \leftarrow b + N(\gamma - T)$
```

normalization (14) of probabilities in kernel logistic regression, the *subtractive* margin normalization (20)-(21) in *Gini*SVM offers several distinct properties in connection with margin based classifiers:

Monotonicity: *Let* $f_k(\mathbf{x}), k = 1,..,M$ *a set of GiniSVM decision functions satisfying the reverse water-filling conditions* $\sum_{k=1}^{M}[f_k(\mathbf{x}) - z_1(\mathbf{x})]_+ = \gamma_1$ *and* $\sum_{k=1}^{M}[f_k(\mathbf{x}) - z_2(\mathbf{x})]_+ = \gamma_2$. *If* $\gamma_1 \geq \gamma_2 > 0$, *then* $z_1(\mathbf{x}) \leq z_2(\mathbf{x})$.

Proof: The two reverse water-filling conditions lead to

$$\sum_{k=1}^{M}\left([f_k(\mathbf{x}) - z_1(\mathbf{x})]_+ - [f_k(\mathbf{x}) - z_2(\mathbf{x})]_+\right) = \gamma_1 - \gamma_2 > 0.$$

The convexity of the hinge function $[a-b]_+ \geq [a]_+ - [b]_+; a, b \in \mathcal{R}$ leads to

$$[z_2(\mathbf{x}) - z_1(\mathbf{x})]_+ \geq (\gamma_1 - \gamma_2)/M > 0$$

which is equivalent to $z_2(\mathbf{x}) > z_1(\mathbf{x})$.

Sparsity: The effect of rectification (20) in the subtractive normalization (21) is to produce a number $0 \leq m < M$ of classes $k_1, \ldots k_m \in \{1, \ldots M\}$ with zero probabilities $P_{k_j}(\mathbf{z}) = 0$, for which the normalization level $z(\mathbf{x})$ exceeds the confidence level $f_{k_j}(\mathbf{x})$. As a direct consequence of the monotonicity property, decreasing the margin parameter γ leads to a larger number m of classes with zero probabilities. Therefore the margin parameter γ directly controls the sparsity m of the probability estimates, assigning the probability mass to a smaller fraction of more confident classes with larger $f_k(\mathbf{x})$ as γ is decreased. Besides the dependence on γ, the number of zero probability classes m depends on the actual values of $f_k(\mathbf{x})$, and hence on the inputs $\mathbf{x}$.

Margin: In the limit $\gamma \rightarrow 0$, the normalization factor $z(\mathbf{x}) \rightarrow max_k f_k(\mathbf{x})$. Thus as $\gamma \rightarrow 0$, margin normalization acts as 'winner-take-all', $m \rightarrow M-1$, and strongly favors the highest class score. Based on this principle a multi-class probability margin can be defined based on the multi-class decision functions $f_k(x)$ as $f_k(\mathbf{x}) = z(\mathbf{x}) + \gamma$. The effect of the hyper-parameter γ can be seen on a synthetic three-class classification problem and is shown in Figure 3. The hyper-parameter γ determines the smoothness of the decision boundary and controls the location of the margin (shown

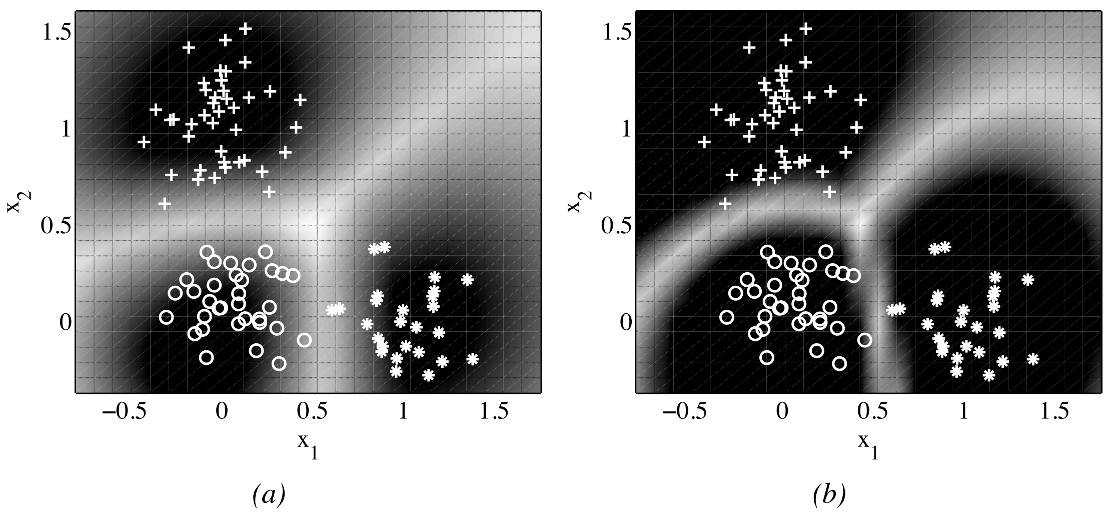


Figure 3: Equal probability contour plots for a three-class problem with the *Gini*SVM solution obtained for (a) $\gamma = 8$ and (b) $\gamma = 0.08$.

by 'white' region). Similar to soft-margin SVM the location of the margin determines the sparsity of the *Gini*SVM solution. This is illustrated in Figure 3(a)-(b). Shades in the figure represent equal probability contours, and the extent of 'white' regions around the decision boundaries illustrates the margin of separation. It can be seen from Figure 3(a)-(b) that reduction in γ has the effect of increasing the size of the margin, and thereby controls the sparsity of the *Gini*SVM solution.

Robustness: The subtractive margin normalization (20) and (21) is inherently robust to impulsive noise, since components $f_k(\mathbf{z})$ in the kernel expansion smaller than a threshold $z(\mathbf{x})$ (at most a margin γ below the largest value) do not contribute to the output.[2] In a physical implementation of margin decoding, adjusting the level γ according to the noise floor leads to significant improvements in decoding performance (Chakrabartty and Cauwenberghs, 2004). The robustness properties of *Gini*SVM in relation to the threshold γ are further analyzed in Section 4.4.

4.2 *Gini*SVM Primal Reformulation

In this section we derive an equivalent primal reformulation of the *Gini*SVM dual (19), analogous to the derivation in Section 3.1. As for the multi-class logistic primal (18), decision functions for classes $k = 1,..,M$ are expressed in terms of a set of M hyperplanes $f_k(\mathbf{x}) = \mathbf{w}_k \cdot \Phi(x) + b_k$. Given a set of training vectors $\mathbf{x}_i \in \mathcal{R}^D, i = 1,..,N$ and its corresponding prior probability distributions $y_{ik} \in \mathcal{R} : y_{ik} \geq 0; \sum_{k=1}^{M} y_{ik} = 1$, *Gini*SVM in its primal reformulation of minimizes a regularization factor proportional to the $L2$ norm of the weight vectors $\mathbf{w}_k, k = 1,..,M$ and a quadratic loss function

2. The subtractive normalization is insensitive only to *negative* impulsive noise in $f_k(\mathbf{z})$. Typically, a choice of kernel indicating a match rather than a mismatch in feature space will avoid positive impulsive noise, since random error is much more likely to activate further mismatch rather than an accidental match.

l_g according to

$$L_g = \frac{1}{2}\sum_{k=1}^{M}|\mathbf{w}_k|^2 + \sum_{i=1}^{N}\sum_{k=1}^{M} l_g(\mathbf{w}_k, b_k, z_i) \tag{22}$$

with loss function $l_g(\mathbf{x}_i)$ for each training vector $\mathbf{x}_i$ given by

$$l_g(\mathbf{x}_i) = \frac{\gamma C}{2}\left(y_{ik} - \frac{1}{\gamma}[f_k(\mathbf{x}_i) - z_i]_+\right)^2,$$

and where $z_i, i = 1, \dots N$ are free parameters entering the minimization of L_g, along with the hyperplane parameters $\mathbf{w}_k$ and $b_k, k = 1, \dots M$.

Proposition II: Denote the solution to the minimization of L_g as

$$(\mathbf{w}_k^*, b_k^*, z_i^*) = \text{argmin}_{\mathbf{w}_k, b_k, z_i} L_g,$$

then

1. $P_k(\mathbf{x}_i) = \frac{1}{\gamma}[f_k^*(\mathbf{x}_i) - z_i^*]_+$ with $f_k^*(\mathbf{x}_i) = \mathbf{w}_k^* \cdot \mathbf{x}_i + b_k^*$ for a given data $\mathbf{x}_i$ is a valid conditional probability measure over classes $k \in 1, .., M$, where z_i performs the normalization $\sum_{k=1}^{M} P_k(\mathbf{x}_i) = 1$.

2. The dual cost function corresponding to the primal cost function (22) is the *Gini*SVM dual (19).

The proof of the proposition is given in Appendix B.

4.3 Binary *Gini*SVM and Quadratic SVM

In one case of interest the multi-class *Gini*SVM solution simplifies to a binary class problem, where the learner is provided with binary labels $y_i \in \{-1, +1\}$ representing class membership of a feature vector $\mathbf{x}_i \in \mathcal{T}$. Binary *Gini*SVM entails regression of a single probability $P_{+1}(\mathbf{x}) = 1 - P_{-1}(\mathbf{x})$ as a function of a single margin variable $f(\mathbf{x}) = \frac{1}{2}(f_{+1}(\mathbf{x}) - f_{-1}(\mathbf{x}))$. Elimination of the normalization parameter z from the binary version of (20) constrained by (21) yields

$$P_{+1}(\mathbf{x}) = \left[\frac{f(\mathbf{x})}{\gamma} + \frac{1}{2}\right]_0^1 \tag{23}$$

where $[.]_0^1$ denotes a limiter function confining the probability to the [0,1] interval, $[a]_0^1 = [a]^+ - [a-1]^+$. With the kernel expansion of $f(\mathbf{x})$ expressed in reduced form[3]

$$f(\mathbf{x}) = \sum_{i=1}^{N} \lambda^i y_i K(\mathbf{x}_i, \mathbf{x}) + b, \tag{24}$$

the *Gini*SVM dual objective function (19) and linear constraints (11) reduce to

$$H_b = \min_{\lambda^i} \frac{1}{C}\left[\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda^i \lambda^j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^{N} G(\lambda^i)\right] \tag{25}$$

3. This choice of kernel expansion is consistent with binary soft-margin SVM and binary KLR, with identical dual formulation under constraints, and with the only difference in the form of the dual potential function $G(\lambda)$ (26).

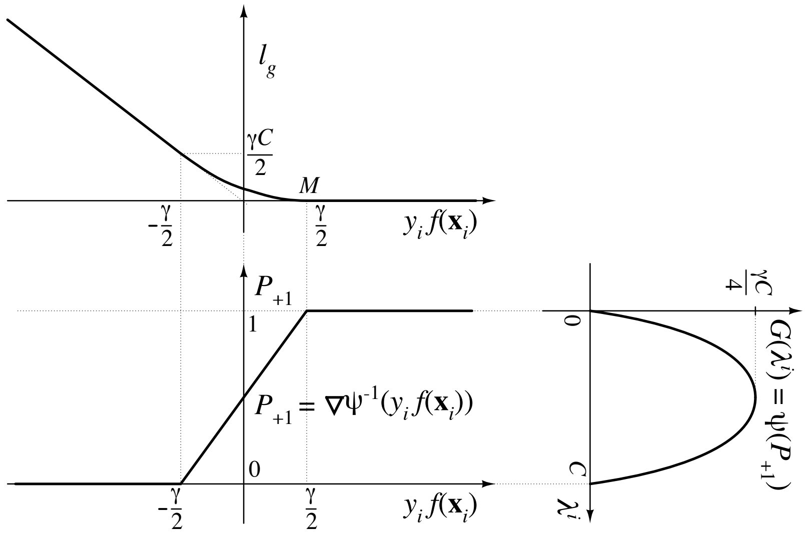


Figure 4: Graphical illustration of the relation between binary *Gini*SVM dual potential function G, inference parameters λ^i, probability estimates P_i, and primal loss function l_g.

under constraints

$$\sum_{i=1}^{M} \lambda^i y_i = 0,$$
$$0 \leq \lambda^i \leq C$$

with dual potential function

$$G(\lambda) = \gamma C \frac{\lambda}{C}(1 - \frac{\lambda}{C}). \tag{26}$$

The derivation is given in Appendix C.

Figure 4 graphically illustrates the relationship between the binary *Gini*SVM dual potential function G, inference parameters λ^i, probability estimates P_i, and primal reformulation loss function l_g. The dual potential is linked to the probability estimate through the Legendre transform $\nabla\Psi^{-1}(.)$ as described by Equations (5)-(7). Since the dual potential function is symmetric $\Psi(P_{+1}) = \Psi(1 - P_{+1}) = \Psi(P_{-1})$, it follows that the Legendre transform $\nabla\Psi^{-1}(.)$ is antisymmetric and hence the probability estimates are centered around the discrimination boundary, $P_{+1}(f(\mathbf{x})) = 1 - P_{+1}(-f(\mathbf{x}))$ consistent with the functional form (23). Symmetry in the dual potential function thus leads to unbiased probability estimates that are centered around the discrimination boundary, $P_{+1} = P_{-1} = 1/2$ for $f(\mathbf{x}) = 0$.

The Legendre transform also links the dual potential function to the primal reformulation cost function. Note the relationship between the parameter γ scaling the dual potential function, and the location of the margin in the loss function and probability estimate indicated by M in Figure 4. Hence the parameter γ can be seen both to control the strength of the agnostic metric D_I, and to control a measure of margin in the probability regression. The regularization parameter C also scales the dual potential function, but controls regularization by scaling the primal loss function by the same factor C without affecting margin, as in soft-margin SVM classification.

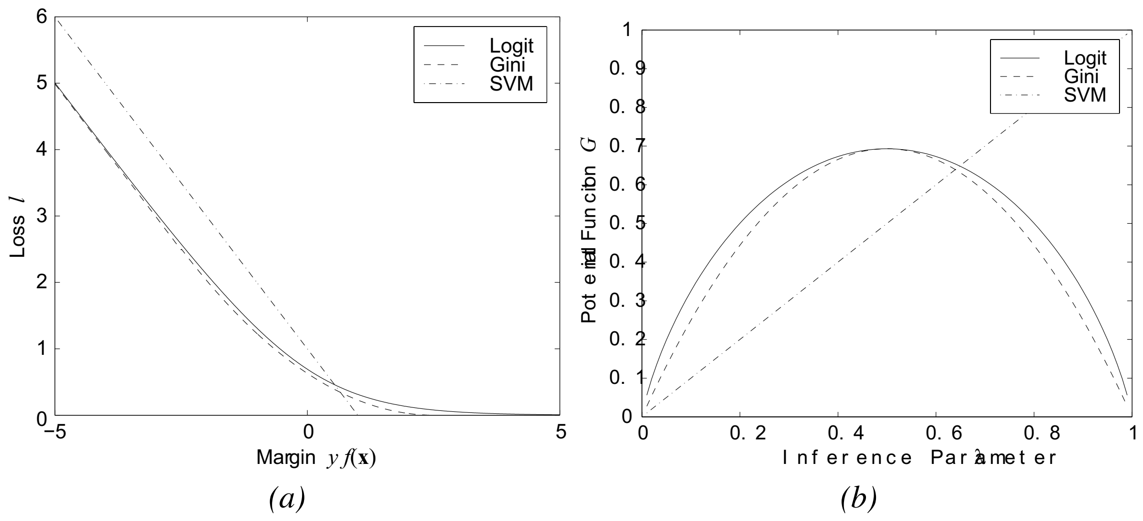


Figure 5: Primal and dual formulation of logistic regression, *Gini*SVM regression and soft-margin SVM classification. (a): Loss function in the primal formulation. (b): Potential function in the dual formulation. The *Gini*SVM loss function and potential function closely approximate those for logistic regression, while offering the sparseness of soft-margin SVM classification. $C = 1$, and $\gamma = 8\log 2$ for *Gini*SVM and $\gamma = 1$ for logistic regression.

A direct comparison can be made between the binary *Gini*SVM formulation and other binary classifiers by inspecting differences in their potential functions $G(u)$, shown in Figure 5(b). The *Gini*SVM potential function is symmetric around the center of the agnostic, uniform distribution $U = 1/2$ where it reaches its maximum. The center corresponds to the origin of the margin variable $y_i f(\mathbf{x}_i)$ in Figure 5(a) which represents the separating hyperplane. Figure 5(b) also shows the binary KLR dual potential function given by Shannon's binary entropy (Jaakkola and Haussler, 1999)

$$G(\lambda) = \gamma C\,\left(\frac{\lambda}{C}\log(\frac{\lambda}{C}) + (1-\frac{\lambda}{C})\log(1-\frac{\lambda}{C})\right).$$

Like *Gini*SVM, the binary KLR dual potential function is symmetric with respect to the separating hyperplane in Figure 5(a), and hence also produces unbiased estimates of conditional probabilities. In contrast, the soft-margin SVM potential function $G(\lambda) = \lambda$ is asymmetric with respect to the separating hyperplane and produces biased or skewed conditional probability estimates. The binary *Gini*SVM dual bears similarity to the quadratic SVM dual (Schölkopf et al., 1998), but the quadratic SVM lacks the symmetry of the potential function around the separating hyperplane.

4.4 Relation to Robust Estimation and Logistic Regression

The *Gini*SVM dual (19) relates to the kernel logistic regression dual (15) through a lower-bound on Shannon entropy. Using the inequality $\log x \leq x - 1, x \geq 0$, the Shannon entropy term $G_e(P) = -\sum_{k=1}^{M} P_k \log P_k$ is everywhere larger than the *Gini* entropy $G_e(P) \geq 1 - \sum_{ik} P_{ik}^2$. This is illustrated in Figure 5(b) which compares the potential functions for KLR and *Gini*SVM in the binary case. Both expressions of entropy reach their maximum for a uniform distribution, $P = \frac{1}{2}$. It can be shown that the solution obtained by minimizing the *Gini*SVM dual H_g in Equation (19) is an over-estimate of the solution obtained by minimizing kernel logistic dual H_e given by Equation (15). In fact we found that initial iterations decreasing the cost H_g also resulted in a decrease of H_e with deviations evident only near convergence. Thus the *Gini*SVM dual H_g can also be used for approximately solving the kernel logistic dual H_e.

The loss function corresponding to binary *Gini*SVM can be visualized through Figure 5(a) and compared with other primals. For binary-class *Gini*SVM, 'margin' is visualized as the extent over which data points are asymptotically normally distributed. Using the notation for asymptotic normality in Huber (1964), the distribution of distance z from one side of the margin for data of one class[4] is modeled by

$$F(z) = (1-\varepsilon)\mathcal{N}(z,\sigma) + \varepsilon\mathcal{H}(z)$$

where $\mathcal{N}(.,\sigma)$ represents a normal distribution with zero mean and standard deviation σ, $\mathcal{H}(.)$ is the unknown symmetrical contaminating distribution, and $0 \leq \varepsilon \leq 1$. $\mathcal{H}(.)$ could, for instance, represent impulsive noise contributing outliers to the distribution.

Huber (1964) showed that for this general form of $F(z)$ the most robust estimator achieving minimum asymptotic variance minimizes the following loss function:

$$g(z) = \begin{cases} \frac{1}{2}\frac{z^2}{\sigma^2} & ; \quad |z| \leq k\sigma \\ k\frac{|z|}{\sigma} - \frac{1}{2}\sigma k^2 & ; \quad |z| > k\sigma \end{cases} \tag{27}$$

where in general the parameter k depends on ε. For *Gini*SVM, the distribution $F(z)$ for each class is assumed one-sided ($z \leq 0$). In particular, the Huber loss function $g(z)$ in (27) reduces to the binary *Gini*SVM loss function $l_g(yf(\mathbf{x}))$, shown in Figure 4, for $z \leq 0$ with $z = yf(\mathbf{x}) - \gamma/2, k\sigma = \gamma$, and $k/\sigma = C$. Therefore the parameter γ in *Gini*SVM can be interpreted as a noise margin in the Huber formulation, consistent with its interpretation as noise threshold in the reverse water-filling procedure for margin normalization. As with soft-margin SVM, points that lie beyond the margin ($z > 0$) are assumed correctly classified, and do not enter the loss function ($g(z) \equiv 0$). The binary *Gini*SVM loss function is a special case of Huber loss used in quadratic SVMs (Schölkopf et al., 1998) which directly generates normalized scores from the margin variable $f(\mathbf{x})$.

5. *Gini*SVM Training Algorithm

*Gini*SVM training entails solving a quadratic optimization problem for which several standard packages and algorithms are available (Platt, 1999b; Cauwenberghs and Poggio, 2001; Osuna et al., 1997). Most of these methods exploit the underlying structure in the classification problem to incorporate heuristics that considerably speed up the convergence of the training algorithm. In this section we describe two algorithms for optimizing *Gini*SVM dual function (19). The first algorithm uses a decomposition algorithm called sequential minimal optimization. The second algorithm uses the polynomial nature of the dual resulting into a novel multiplicative update algorithm based on growth transformation on probabilities.

4. The distributions for the two classes are assumed symmetrical, with margin on opposite sides, and distance z in opposite directions.

5.1 Sequential Minimal Optimization

Sequential minimal optimization (Platt, 1999b) is an extreme case of a decomposition based quadratic program solver, where a smallest set of inference parameters is chosen each iteration, and optimized subject to the linear constraints. The advantage of SMO is that it can be efficiently implemented without resorting to QP packages, and it scales to very large data sets. In the case of the *Gini*SVM dual function (19), at least four inference parameters need to be chosen to satisfy two sets of equality constraints (11). A randomized version of SMO algorithm is described in Algorithm 2.

Algorithm 2 Randomized SMO algorithm

```
Require: Training data $\mathbf{x}_i, i = 1,..,N$ and labels $y_{ik}, i = 1,..,N, k = 1,..,M$
Ensure: Let $\lambda_k^i = 0$ for $i = 1,..,N, k = 1,..,M$.
  repeat
    • Randomly choose a set of four inference parameters $\lambda_1^1, \lambda_2^1, \lambda_1^2$ and $\lambda_2^2$.
    • Update $\lambda_1^1, \lambda_2^1, \lambda_1^2$ and $\lambda_2^2$ such that the dual (19) is minimized subject to constraints (11).
  until convergence
```

The derivation of the SMO update rule based on the choice of inference parameters is given in Appendix D. Instead of random selection of working sets of inference parameters, heuristics based on the structure of the classification problem can be used to speed up convergence (Platt, 1999b; Keerthi et al., 2001). Standard QP algorithmic methods for SVM training such as caching and shrinking besides chunking (Joachims, 1998) can be applied to further speed up convergence of SMO training.

5.2 Growth Transformation on Generalized Polynomial Dual

In lieu of the inference parameters defined as $\lambda_k^i = C[y_{ik} - P_k(\mathbf{x}_i)]$, the *Gini*SVM dual in (19) can be expressed in terms of probabilities $P_{ik} = P_k(\mathbf{x}_i)$ as

$$H = \frac{C}{2}\sum_{k=1}^{M}\left[\sum_{i=1}^{N}\sum_{j=1}^{N} Q_{ij}\left[y_{ik} - P_{ik}\right]\left[y_{jk} - P_{jk}\right] + \frac{\gamma}{2}\sum_{i=1}^{N} P_{ik}^2\right] \tag{28}$$

with linearity constraints (3) and (4) to ensure valid probabilities, $P_{ik} \geq 0, \forall i,k$ and $\sum_{k=1}^{M} P_{ik} = 1, \forall i$. For the remainder of the derivation the additional equality constraint (2) corresponding to the bias term b will be relaxed. Artifacts due to absence of the bias b can reduced by properly pre-processing and centering the training data or by incorporating an additional input dimension in the kernel function. The optimization function (28) is a non-homogeneous polynomial with normalized probability variables $P_{ik}, \forall i$ and with possibly negative coefficients. We can directly apply results from Baum and Sell (1968) and Gopalakrishnan et al. (1991) to optimize the dual (28).

Theorem 2 (Gopalakrishnan et al.) Let $H(\{P_{ik}\})$ a polynomial of degree d in variables P_{ik} in the domain $D : P_{ik} \geq 0, \sum_{k=1}^{q_i} P_{ik} = 1, i = 1,..,N, k = 1,..,q_i$ such that $\sum_{k=1}^{q_i} P_{ik}\frac{\partial H}{\partial P_{ik}}(P_{ik}) / \neq 0 \quad \forall i$. Define an iterative map according to the following recursion

$$\widehat{P}_{ik} \leftarrow \frac{P_{ik}(\frac{\partial H}{\partial P_{ik}}(P_{ik}) + \Gamma)}{\sum_{k=1}^{q_i} P_{ik}(\frac{\partial H}{\partial P_{ik}}(P_{ik}) + \Gamma)} \tag{29}$$

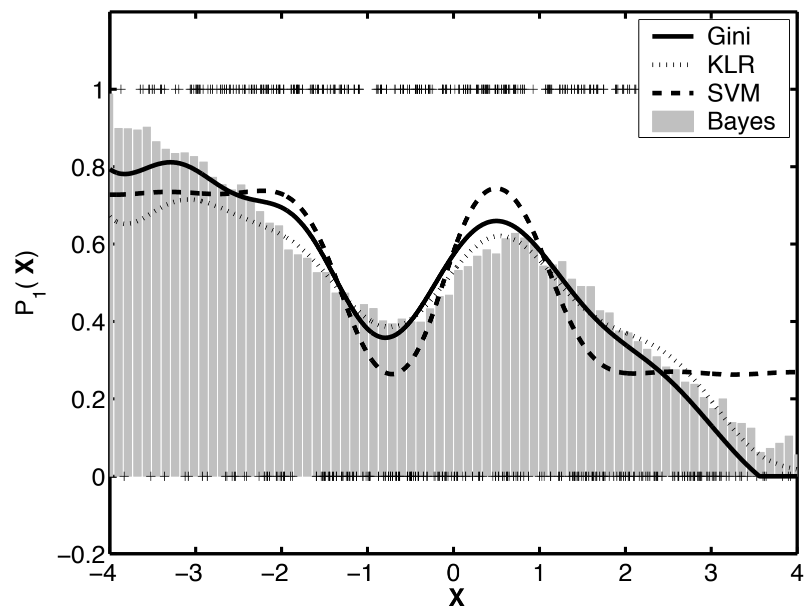


Figure 6: Probability estimates generated by *Gini*SVM, KLR and a calibrated soft-margin SVM for one-dimensional synthetic training data.

where $\Gamma \geq Sd(N+1)^{d-1}$ with S being the smallest coefficient of the polynomial $H(\{P_{ik}\})$. Then $\{\widehat{P}_{ik}\} \in D$ and $H(\{\widehat{P}_{ik}\}) > H(\{P_{ik}\})$.

The result can be applied for minimizing the polynomial dual corresponding to Equation (29). Let $P^0_{ik} = 1/M$ the initial value of the probability distribution for all i,k, and assume the kernel matrix be bounded such that $|Q_{ij}| \leq Q_{max}, \forall i,j$. Also, let P^m_{ik} the value of the probability distribution at m^{th} iteration then

$$P^{m+1}_{ik} \leftarrow P^m_{ik}\delta^m_{ik} / \sum_{k=1}^{M} P^m_{ik}\delta^m_{ik}$$

where

$$\delta^m_{ik} = C\sum_{j=1}^{N} Q_{ij}\,[P^m_{ik} - y_{ik}] + \gamma P^m_{ik} + \Gamma$$

and $\Gamma = C\ (N+1)\ Q_{max}$. At each update the cost function (28) decreases, and the procedure is repeated till convergence. Due to the multiplicative nature of the update some distribution variables P_{ik} can never reach unity or zero; however, in practice it approaches the limits within given margins of precision similar to other implementations of SVM training algorithms. As with other SVM optimization techniques, the speed of large margin growth transformation can be enhanced by using caching and shrinking (Joachims, 1998), as values of the distribution P_{ik} close to unity or zero almost do not change.

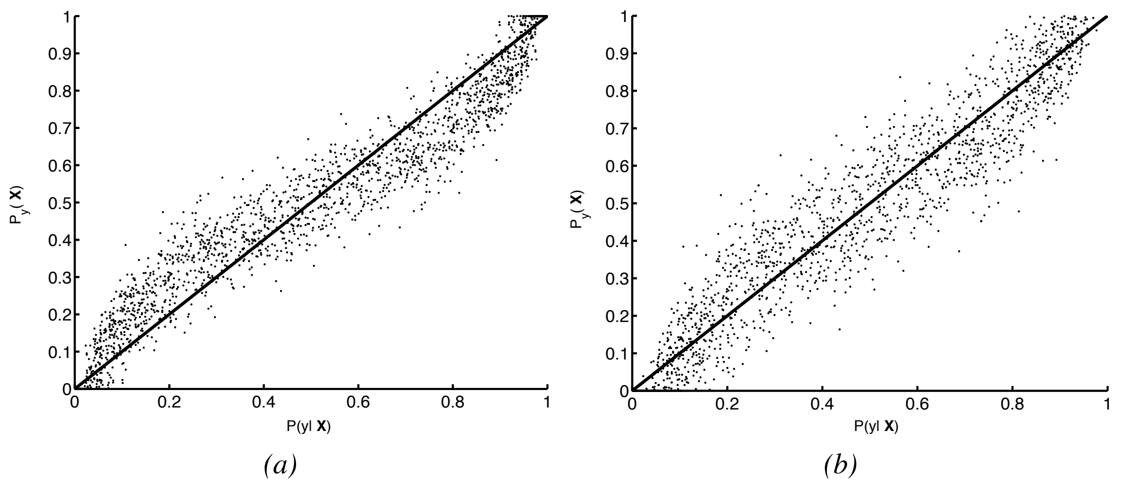


Figure 7: Comparison between conditional Bayes probability estimates and scores generated by *Gini*SVM for 10-dimensional synthetic data with *(a)* $\gamma = 0.8$ and *(b)* $\gamma = 0.08$.

6. Experiments and Results

The first set of experiments were designed to characterize the probability scores generated by *Gini*SVM for synthetic data. Figure 6 compares the scores generated by *Gini*SVM, KLR and soft-margin SVM for a synthetic binary classification problem. The one dimensional training data corresponding to two classes were generated using a bimodal Gaussian distribution. A histogram generated by data points randomly sampling the distribution is shown in Figure 6 and the locations of 500 data points used for training are denoted by '+' along $y = 1$ and $y = 0$. For the soft-margin SVM the scores were normalized using Platt's calibration procedure (Platt, 1999a). The Figure 6 shows that the scores generated by KLR, *Gini*SVM and calibrated soft-margin SVM are similar and approximate the sampled distribution which approximates the Bayesian optimum solution. It can be seen that calibrated soft-margin SVM scores do not approximate the true conditional distribution at the boundary of the distribution and would require additional parameterization for producing better estimates.

Figures 7(b) and (c) compare *Gini*SVM scores with sampled conditional scores (Bayes estimates) for synthetic data in 10 dimensions. The data were generated from a multi-variate Gaussian distribution, out of which 100 data points were chosen for training. Figures 7(a) and (b) demonstrate a monotonic relationship between *Gini*SVM scores and Bayes estimate of class conditional probabilities. The sigmoidal relationship trend shown in the scatter plot 7(a) for $\gamma = 0.8$ is attributed to linear approximation of the logistic model (14) by subtractive normalization model (20).

The performance of *Gini*SVM based classifier was evaluated on three benchmark UCI databases and compared with a baseline one-vs-all soft-margin SVM classification method (Weston and Watkins, 1998; Crammer and Singer, 2000). Table 1 summarizes the results obtained for the *Gini*SVM classifier. Data sets are labeled with attributes as (N, D, M) where N denotes its total size, D denotes the dimension of the input vector and M denotes the total number of classes. All training data were normalized between $[-1, 1]$ and a 10-fold cross validation procedure was used to obtain average classification error rate and average number of support vectors. A Gaussian

Iris (150,3,4)			**Ecoli (336,8,7)**			**Glass (214,6,13)**		
(γ, C)	**Err(%)**	**nsv(%)**	(γ, C)	**Err(%)**	**nsv(%)**	(γ, C)	**Err(%)**	**nsv(%)**
$(0.8, 0.5)$	3.2 ± 2	40 ± 1.6	(0.8,1)	14 ± 3	60 ± 5	(0.8,1)	30 ± 4.7	95 ± 1.2
$(0.8, 5)$	4.0 ± 2	19 ± 2.2	(0.8,10)	15 ± 1.8	62 ± 7	(0.8,10)	29 ± 3	89 ± 1.5
$(0.08, 5)$	4.8 ± 2	14 ± 3	(0.08,1)	12.7 ± 2.7	63 ± 7	(0.08,10)	32 ± 6	82 ± 2.6
Baseline one-vs-all SVM								
(C = 4)	3.4 ± 3	18 ± 1	(C = 10)	14 ± 4	61 ± 6	(C = 5)	30 ± 2	81 ± 3

Table 1: Performance of *Gini*SVM classifier on UCI database.

kernel $K(\mathbf{x}, \mathbf{y}) = \exp(-\frac{1}{2\sigma^2}(\mathbf{x}-\mathbf{y})^T(\mathbf{x}-\mathbf{y}))$ was chosen for all experiments. The kernel parameter σ and the regularization parameter C were chosen based on the performance of the baseline SVM on a held-out set. The same kernel parameter was used for training *Gini*SVM classifiers. Table 1 shows the error rate (indicated by **Err**) and the number of support/error vectors (indicated by **nsv**) obtained for different sets of hyper-parameters γ and C. The results indicate that the classification performance of the *Gini*SVM based system is comparable to the baseline one-vs-all SVM system. The results also illustrate the effect of γ on the sparseness of the solution which increases as $\gamma \rightarrow 0$, as explained using the generalized dual framework in Section 4.

6.1 Face Detection and Effects of Parameter Mismatch

The advantage of *Gini*SVM over conventional soft-margin SVM is demonstrated by performing sensitivity analysis on the kernel expansion at completion of training. For this experiment a face detection task was chosen. The classifiers were trained using the face detection database available through CBCL at MIT (Alvira and Rifkin, 2001) and their performance was evaluated on the standard CMU-MIT test set (Rowley et al., 1998). Training of the classifier was performed by utilizing floating point precision arithmetic, whereas evaluation was performed after quantizing the support vectors and inference parameters to $8, 6$ and 4 bits, and adding 1 LSB of uniform random noise. For this experiment the parameter C was determined by optimizing the performance of the classifier on a held-out data set. Receiver operating characteristics (ROC) were obtained by evaluating the performance of the mismatched classifier on the test set. Figure 8 compares ROC curves for the classifier trained with soft-margin SVM, vs. another trained identically with *Gini*SVM for a 2^{nd} order polynomial kernel. The results indicate that *Gini*SVM solution is more robust to mismatch and precision errors in the inference parameters. In fact for this data set, the *Gini*SVM solution quantized to 1 bit is more robust than an equivalent soft-margin SVM solution quantized to 4 bits.

6.2 Speaker Verification Experiments

The benefit of normalized scores generated by *Gini*SVM is demonstrated for the task of text-independent speaker verification. The task entails verifying a particular speaker from possible imposters without any knowledge of the text spoken. A conventional approach uses a classifier to generate scores based on individual speech frames. The scores are integrated over the duration of the utterance and compared against a threshold to accept or reject the speaker. A YOHO speaker verification database was chosen for training and testing the speaker verification system. The YOHO database consists of sets of 4 combination lock phrases spoken by 168 speakers. For each utterance

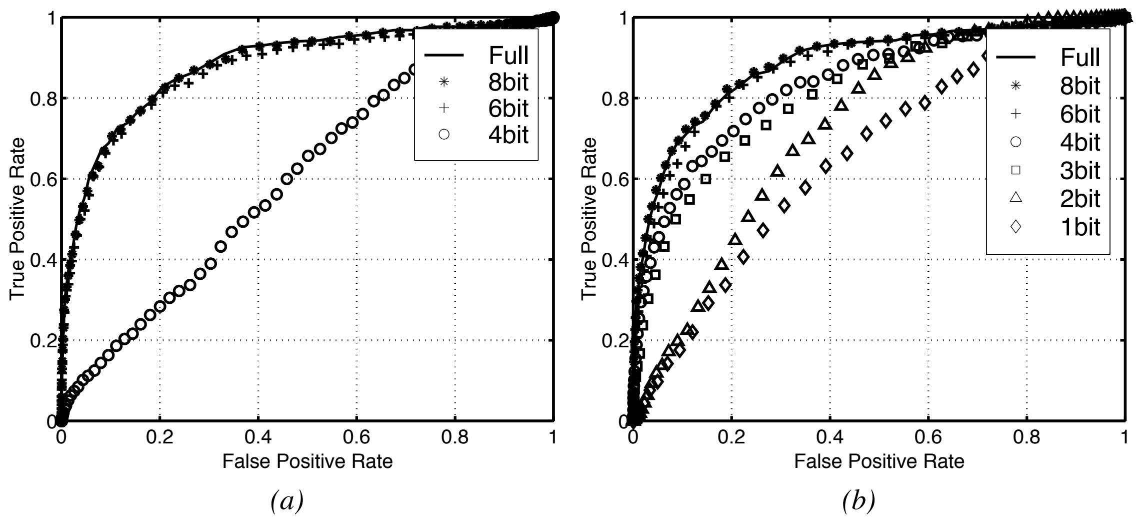


Figure 8: ROC obtained for a face detection system trained with *(a)*: soft-margin SVM and *(b)*: *Gini*SVM training algorithm, for a 2^{nd} order polynomial kernel.

contiguous 25ms speech samples were extracted and Mel-frequency cepstral coefficient (MFCC) features were extracted. The MFCC feature extraction procedure has been extensively studied in the literature and details can be found in Rabiner and Juang (1993). A 39-dimensional feature vector was formed by concatenating the total energy in the speech frame, along with the Δ and $\Delta - \Delta$ MFCC coefficients.

For training, 100 speakers (speaker ID: 101-200) were chosen from the YOHO database and MFCC features were extracted for all speech frames corresponding to each speaker. To reduce the total number of training points, a K-means clustering was performed for each speaker to obtain 1000 cluster points for the correct speaker, and 100 cluster points for each imposter speaker. For each speaker (101-200), this procedure was repeated to obtain a training set of $10,900$ MFCC vectors. Classifiers specific to each speaker were trained using a *Gini*SVM toolkit (http://bach.ece.jhu.edu/svm/ginisvm). For testing utterances corresponding to 100 speakers were chosen from the YOHO test set. Confidence scores generated by *Gini*SVM for each speech frame were integrated over the duration of the utterance to obtain the final cumulative score. Thus each speech frame is treated to be independent and their scores are integrated together without taking into account any time-based correlations.

Figure 9 compares the ROC obtained by a soft-margin SVM based system with a *Gini*SVM based verification system trained for one speaker (id: 148). The speaker with worst verification performance among all was selected. Figure 9 shows that a *Gini*SVM based system exhibits better verification performance compared to an equivalent soft-margin SVM. For each ROC (one per speaker) an equal error rate (EER) parameter was computed. The EER metric is widely used for quantifying performance of a biometric system and is defined as the error rate at which total false positive rate is equal to false rejection rate. Thus, the lower the EER, the more robust is the performance of a biometric system. For this experiments EERs corresponding to each speaker verification system (101-200) were averaged to obtain an equivalent system EER. For a soft-margin SVM and

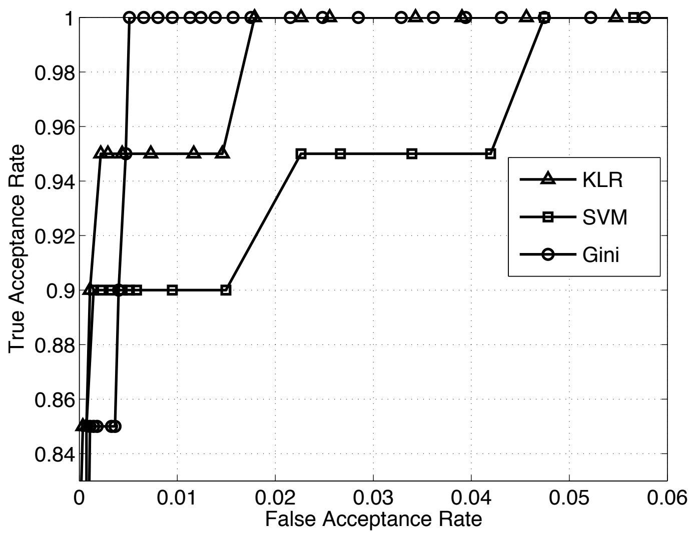


Figure 9: Comparison of ROC obtained for a speaker verification system based on soft-margin SVM and *Gini*SVM classification for speaker id: 148.

KLR, the average EER was computed to be equal to 0.36% and 0.35%, where as the EER for a *Gini*SVM based system was found to 0.28%. This demonstrates that the normalization procedure used by *Gini*SVM improves the accuracy of a text-independent speaker verification system. The verification results are also comparable with other reported results on the YOHO data set (Campbell et al., 2002).

7. Conclusions and Extensions

We introduced a general, maximum entropy based framework for constructing multi-class support vector machines that generate normalized scores. In particular, *Gini*SVM produces direct estimates of conditional probabilities that approximate kernel logistic regression (KLR) at reduced computational cost, incurring quadratic programming under linear constraints as with standard SVM training. Unlike a baseline soft-margin SVM based system with calibrated probabilities, *Gini*SVM produces unbiased probability estimates owing to symmetry in the agnostic distance metric in the maximum entropy formulation. The probability estimates are sparse, where the number of non-zero probabilities is controlled by a single parameter γ, which acts as a margin in the normalization of probability scores. The margin parameter γ is distinct from the regularization parameter C also found in soft-margin SVM and KLR, even though both C and γ weigh the agnostic metric relative to the prior metric in the maximum entropy primal cost function.

For efficient implementation of *Gini*SVM training, we presented a modified sequential minimum optimization (SMO) algorithm, and a multiplicative update algorithm based on growth transformation on probability functions in the dual. The performance of *Gini*SVM probability regression and classification was evaluated on benchmark UCI databases in comparison with KLR and soft-margin SVM. Results on face detection database indicated that the solution obtained by *Gini*SVM training is more robust to mismatch in inference parameters, offering advantages in efficient, reduced precision implementation of SVMs. *Gini*SVM also successfully trained on a task of text-independent speaker verification, by integrating normalized probability scores over time. *Gini*SVM further extends to forward decoding kernel machines for trainable dynamic probabilistic inference on graphs (Chakrabartty and Cauwenberghs, 2002).

The maximum entropy framework for large-margin kernel probability regression introduced for *Gini*SVM is general and can be extended to other classification and regression tasks based on polynomial entropy. Of particular interest are formulations that use symmetric potential functions like the Gini quadratic entropy function.

Acknowledgments

This work was supported by grants from the Catalyst Foundation (*http://www.catalyst-foundation.org*), the National Science Foundation, and the Office of Naval Research.

Appendix A. Kernel Logistic Regression Primal and Dual Formulation

Proof of Proposition I: Define L_e as the regularized log-likelihood/cross entropy for kernel logistic regression (Wahba, 1998; Zhu and Hastie, 2002)

$$L_e = \sum_{k=1}^{M} \frac{1}{2}||\mathbf{w}_k||^2 - C\sum_{i=1}^{N}[\sum_{k=1}^{M} y_{ik} f_k(\mathbf{x}_i) - \log(e^{f_1(\mathbf{x}_i)} + ... + e^{f_M(\mathbf{x}_i)})] \ . \tag{30}$$

First order conditions with respect to parameters $\mathbf{w}_k$ and b_k in $f_k(\mathbf{x}) = \mathbf{w}_k.\mathbf{x} + b_k$ yield

$$\begin{aligned} \mathbf{w}_k &= C\sum_{i=1}^{N}[y_{ik} - \frac{e^{f_k(\mathbf{x}_i)}}{\sum_p^M e^{f_p(\mathbf{x}_i)}}]\ \mathbf{x}_i, \\ 0 &= C\sum_{n}^{N}[y_{ik} - \frac{e^{f_k(\mathbf{x}_i)}}{\sum_p^M e^{f_p(\mathbf{x}_i)}}] \ . \end{aligned} \tag{31}$$

Denote

$$\lambda_k^n = C[y_{ik} - \frac{e^{f_k(\mathbf{x}_i)}}{\sum_p^M e^{f_p(\mathbf{x}_i)}}] \tag{32}$$

in the first-order conditions (31) to arrive at the kernel expansion (17) with linear constraint

$$\begin{aligned} f_k(\mathbf{x}) &= \sum_n \lambda_k^n K(\mathbf{x}_i, \mathbf{x}) + b_k, \\ 0 &= \sum_n \lambda_k^n \ . \end{aligned} \tag{33}$$

Note also that $\sum_{k=1}^{M} \lambda_k^n = 0$ by construction.

Legendre transformation of the primal objective function (30) in $\mathbf{w}_k$ and b_k leads to a dual formulation directly in terms of the coefficients λ_k^n (Jaakkola and Haussler, 1999). Define $z_n = \log(\sum_p^M e^{f_p(\mathbf{x}_i)})$, and $Q_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$. Then (32) and (33) transform to

$$\sum_l Q_{nl}\lambda_k^l - \log[y_{nk} - \lambda_k^n/C] + b_k - z_n = 0$$

which correspond to first-order conditions of the convex dual functional

$$H_e = \sum_{k=1}^{M} [\frac{1}{2}\sum_{n}^{N}\sum_{l}^{N}\lambda_k^n Q_{nl}\lambda_k^l + C\sum_{n}^{N}(y_{nk}-\lambda_k^n/C)\log(y_{nk}-\lambda_k^n/C)]$$

under constraints

$$\begin{aligned}\sum_n \lambda_k^n &= 0, && (34)\\ \sum_k \lambda_k^n &= 0, && (35)\\ \lambda_k^n &\le Cy_{nk}\end{aligned}$$

where b_k and z_n serve as Lagrange parameters for the equality constraints (34) and (35).

Appendix B. *GiniSVM* Primal and Dual Formulation

Proof of Proposition II: The hinge function $[.]_+$ can be appropriately modeled by introducing slack variables $\mu_{ik} \geq 0$ into the loss function such that

$$L_g(\mathbf{w}_k, b_k, z_i) = \min_{\mu_{ik}\geq 0} \frac{1}{2}\sum_k |\mathbf{w}_k|^2 + \frac{\gamma C}{2}\sum_{ik}(y_{ik} - \frac{1}{\gamma}[f_k(\mathbf{x}_i) - z_i + \mu_{ik}])^2 + \sum_{ik}\eta_{ik}\mu_{ik}$$

The first order conditions corresponding to the variables $\mathbf{w}_k, b_k, z_i, \mu_{ik}$ are given by

$$\begin{aligned}\partial F/\partial \mathbf{w}_k &= \mathbf{w}_k - C\sum_{i=1}^{N}\left(y_{ik} - \frac{1}{\gamma}[f_k(\mathbf{x}_i) - z_i + \mu_{ik}]\right)\mathbf{x}_i = 0, && (36)\\ \partial F/\partial b_k &= C\sum_{i=1}^{N}\left(y_{ik} - \frac{1}{\gamma}[f_k(\mathbf{x}_i) - z_i + \mu_{ik}]\right) = 0, && (37)\\ \partial F/\partial z_i &= C\sum_{k}\left(y_{ik} - \frac{1}{\gamma}[f_k(\mathbf{x}_i) - z_i + \mu_{ik}]\right) = 0 && (38)\\ \partial F/\partial \mu_{ik} &= \eta_{ik} - \frac{1}{\gamma}[f_k(\mathbf{x}_i) - z_i + \mu_{ik}] = 0 && (39)\end{aligned}$$

where η_{ik} are the Lagrange multipliers corresponding to the inequality conditions $\mu_{ik} \geq 0$. The complementary slackness criterion (Bertsekas, 1995) for these constraints gives $\eta_{ik} \geq 0$ and $\eta_{ik}\mu_{ik} = 0$ which along with criterion (39) gives

$$\frac{1}{\gamma}[f_k(\mathbf{x}_i) - z_i + \mu_{ik}] = \frac{1}{\gamma}[f_k(\mathbf{x}_i) - z_i]_+ \geq 0 \qquad (40)$$

and, according to (38),

$$\sum_k \frac{1}{\gamma}[f_k(\mathbf{x}_i) - z_i]_+ = 1$$

which proves the first part of the proposition.

To prove the second part of the proposition let

$$\lambda_k^i = Cy_{ik} - \frac{1}{\gamma}[f_k(\mathbf{x}_i) - z_i + \mu_{ik}]). \qquad (41)$$

Criteria (37), (38) and (40) lead to the constraints (11). Substitution in (36) yields an expansion of $\mathbf{w}_k$ which re-substituted in the primal yields the dual first-order condition

$$\sum_j Q_{ij}\lambda_k^j + b_k - z_i + \mu_k^i - \gamma(y_{ik} - \lambda_k^i/C) = 0.$$

Along with constraints (11) the corresponding dual reduces to the *Gini*SVM dual H_g (19), which completes proof of the proposition.

Appendix C. Binary *Gini*SVM Dual Formulation

For a binary *Gini*SVM the dual cost function (19) becomes

$$H_g = \frac{1}{2C}\sum_{ij}\lambda_{+1}^i\lambda_{+1}^j Q_{ij} + \frac{1}{2C}\sum_{ij}\lambda_{-1}^i\lambda_{-1}^j Q_{ij} + \frac{\gamma}{2}\sum_i (y_{i,+1} - \lambda_{+1}^i/C)^2 + \frac{\gamma}{2}\sum_i (y_{i,-1} - \lambda_{-1}^i/C)^2 \quad (42)$$

and the constraints (11) are written as

$$\lambda_{-1}^i = -\lambda_{+1}^i, \quad (43)$$

$$\sum_{i=1}^N \lambda_{-1}^i = \sum_{i=1}^N \lambda_{+1}^i = 0, \quad (44)$$

$$\lambda_{+1}^i \le C y_{i,+1}, \quad (45)$$

$$\lambda_{-1}^i \le C y_{i,-1}. \quad (46)$$

Let $\lambda^i = y_i\lambda_{+1}^i$ where $y_i = (2y_{i,+1} - 1)$. Then $f(\mathbf{x}) = \frac{1}{2}(f_{+1}(\mathbf{x}) - f_{-1}(\mathbf{x}))$ reduces to the kernel expansion (24) with $b = \frac{1}{2}(b_{+1} - b_{-1})$. For binary labels $\mathbf{y}_i = \pm 1$ the equality and inequality constraints (43)-(46) simplify to

$$\sum_{i=1}^N \lambda^i y_i = 0,$$

$$0 \le \lambda^i \le C.$$

Further substitution of $\lambda_{+1}^i = y_i\lambda^i$, $\lambda_{-1}^i = -y_i\lambda^i$, $y_{i,+1} = \frac{1}{2}(1+y_i)$ and $y_{i,-1} = \frac{1}{2}(1-y_i)$ into the binary *Gini*SVM dual cost function (42)

$$H_g = \frac{1}{C}\sum_{ij}\lambda^i\lambda^j y_i y_j Q_{ij} - \gamma\sum_{i=1}^N\left(\left(\frac{1}{2}\right)^2 - \left(\frac{1}{2} - \frac{\lambda^i}{C}\right)^2\right)$$

which is equivalent to the form H_b (25).

Appendix D. *Gini*SVM Sequential Minimum Optimization

The following extends the original SMO algorithm (Platt, 1999b) from binary soft-margin SVM to multi-class *Gini*SVM.

Let $\lambda_k^{i*} \in C$ be a set of parameters in the constraint space C given by (11). Each iteration a set of four inference parameters, indexed by class identifiers k_1, k_2 and data identifiers i_1, i_2, are jointly updated. Without loss of generality the parameters of this working set will be referred to as $\lambda_1^{1*}, \lambda_2^{1*}, \lambda_1^{2*}$ and λ_2^{2*}, where the indices correspond to k_1, k_2 and i_1, i_2. The aim of an SMO update is to find a new estimate of these coefficients $\lambda_1^1, \lambda_2^1, \lambda_1^2$ and λ_2^2 such that the new set of coefficients affect a net decrease in the objective function H_g (19), while still satisfying constraints C (11). This is ensured by

$$\begin{aligned}
\lambda_1^1 + \lambda_2^1 &= \zeta_1 = \lambda_1^{1*} + \lambda_2^{1*} \\
\lambda_1^1 + \lambda_1^2 &= \xi_1 = \lambda_1^{1*} + \lambda_1^{2*} \\
\lambda_2^1 + \lambda_2^2 &= \xi_2 = \lambda_2^{1*} + \lambda_2^{2*} \\
\lambda_2^2 + \lambda_1^2 &= \zeta_2 = \lambda_2^{2*} + \lambda_1^{2*}.
\end{aligned}$$

Only three of the above equalities need to be satisfied as the fourth one is automatically satisfied. Decomposing the *Gini*SVM dual in terms of these four coefficients leads to

$$\begin{aligned}
H &= \frac{1}{2}Q_{11}(\lambda_1^1)^2 + Q_{12}\lambda_1^1\lambda_1^2 + \frac{1}{2}Q_{22}(\lambda_1^2)^2 + \lambda_1^1 \sum_{j \neq 1,2} Q_{1j}\lambda_1^j + \lambda_1^2 \sum_{j \neq 1,2} Q_{2j}\lambda_1^j \\
&+ \frac{1}{2}Q_{11}(\lambda_2^1)^2 + Q_{12}\lambda_2^1\lambda_2^2 + \frac{1}{2}Q_{22}(\lambda_2^2)^2 + \lambda_2^1 \sum_{j \neq 1,2} Q_{1j}\lambda_2^j + \lambda_2^2 \sum_{j \neq 1,2} Q_{2j}\lambda_2^j \\
&+ \gamma C(y_{11} - \lambda_1^1/C)^2 + \gamma C(y_{21} - \lambda_1^2/C)^2 + \gamma C(y_{12} - \lambda_2^1/C)^2 + \gamma C(y_{22} - \lambda_2^2/C)^2.
\end{aligned}$$

Substituting

$$\begin{aligned}
\lambda_2^1 &= \zeta_1 - \lambda_1^1, \\
\lambda_1^2 &= \xi_1 - \lambda_1^1, \\
\lambda_2^2 &= \xi_2 - \zeta_1 + \lambda_1^1
\end{aligned}$$

and using the first order condition $\partial H/\partial\lambda_1^1 = 0$, optimal values for λ_1^{1*} are found as

$$\lambda_1^{1*} = \lambda_1^1 + (g_{12} + g_{21} - g_{11} - g_{22})/2\eta \tag{47}$$

where

$$g_{lm} = -2\gamma y_{lm} + \sum_j Q_{lj}\lambda_m^j + 2\gamma/C\lambda_m^l$$

and

$$\eta = Q_{11} + Q_{22} - 2Q_{12} + 4\gamma/C.$$

At each step of the update (47) the *Gini*SVM dual function decreases, and repeated sampling of the four-point working set over the training set ensures proper convergence to the true minimum, barring degeneracies in the cost function. At convergence the parameters $b_k, k = 1,..,M$ are obtained by solving a set of overcomplete equations for data points that lie in the interior of the boundary constraints $\lambda_k^i < Cy_{ik}$. For the interior points denoted by its training index i the following condition is satisfied

$$b_k - z_i + g_{ik} = 0$$

which is over-complete in parameters $b_k, k = 1,..,M$ and $z_i, i = 1,..,I$, where I denotes the total number of training points within the interior of the constraints.

References

E.L. Allwein, R.E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research,* 1:113-141, 2000.

M. Alvira and R. Rifkin. An empirical comparison of SNoW and SVMs for face detection. *CBCL paper 193 /AI Memo 2001-2004*, MIT, 2001.

R. Auckenthaler, M. Carey and H. Lloyd-Thomas. Score normalization for text-independent speaker verification system. *Digital Signal Processing,* 10(1):42-54, 2000.

L.E. Baum and G. Sell. Growth transformations for functions on manifolds. *Pacific J. Math.*, 27(2):211-227, 1968.

D. Bertsekas. *Non-linear Programming*. Athena Scientific, MA, 1995.

B. Boser, I. Guyon and V. Vapnik. A training algorithm for optimal margin classifier. *Proc. 5th Ann. ACM Workshop on Computational Learning Theory (COLT)*, pages 144-52, 1992.

L. Breiman, J.H. Friedman and R. Olshen. *Classification and Regression Trees*. Wadsworth and Brooks, Pacific Grove CA, 1984.

C. Burges. A tutorial on support vector machines for pattern recognition. U. Fayyad, Ed., *Proc. Data Mining and Knowledge Discovery*, pages 1-43, 1998.

W.M. Campbell, K.T. Assaleh and C.C. Broun. Speaker with polynomial classifiers. *IEEE Trans. Speech and Audio Proc.,* 10(4):205-212, May 2002.

G. Cauwenberghs and T. Poggio. Incremental and decremental support vector machine learning. *Adv. Neural Information Processing Systems 10*, Cambridge MA: MIT Press, 2001.

S. Chakrabartty and G. Cauwenberghs. Forward decoding kernel machines: A hybrid HMM/SVM approach to sequence recognition. *IEEE Int. Conf. of Pattern Recognition: SVM workshop. (ICPR'2002)*, 2002.

S. Chakrabartty and G. Cauwenberghs. Margin propagation and forward decoding in analog VLSI. *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS'2004)*, 2004.

S. Chakrabartty and G. Cauwenberghs. Sub-microwatt analog VLSI support vector machine for pattern classification and sequence estimation. *Adv in Neural Information Processing Systems 17*, Cambridge: MIT Press, 2005.

T.M. Cover and J.A. Thomas, *Elements of Information Theory.* John Wiley and Sons, 1991.

K. Crammer and Y. Singer. The learnability and design of output codes for multiclass problems. *Proc. 13th Ann. Conf. Computational Learning Theory (COLT),* 2000.

T.G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263-286, 1995.

F. Girosi, M. Jones and T. Poggio. Regularization theory and neural networks architectures. *Neural Computation*, 7:219-269, 1995.

P.S. Gopalakrishnan, D. Kanevsky, A. Nadas and D. Nahamoo. An inequality for rational functions with applications to some statistical estimation problems. *IEEE Trans. Information Theory*, 37(1):107-113, 1991.

Y. Gu and T. Thomas. A text-independent speaker verification system using support vector machines classifier. *Proc. Eur. Conf. Speech Communication and Technology (Eurospeech'01),* pages 1765-1769, 2001.

C. Hsu and C. Lin. A comparison of methods for multi-class support vector machines. *IEEE Trans. Neural Networks,* 13(2):415-425, 2002.

P.J. Huber. Robust Estimation of Location Parameter. *Annals of Mathematical Statistics*, volume 35, 1964.

T. Jaakkola and D. Haussler. Probabilistic kernel regression models. *Proc. 7th Int. Workshop on Artificial Intelligence and Statistics*, 1999.

E. Jaynes. Information theory and statistical mechanics. *Physics Review*, 106:620-630, 1957.

T. Jebara. Discriminative, generative and imitative learning. PhD Thesis, MIT Media Laboratory, 2001.

T. Joachims. Text categorization with support vector machines. Technical Report LS-8 23, Univ. of Dortmund, 1997.

T. Joachims. Making large-scale support vector machine learning practical, In Schölkopf, Burges and Smola, Eds., *Advances in Kernel Methods: Support Vector Machines,* Cambridge MA: MIT Press, 1998.

M.I. Jordan and R.A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Proc. Int. Joint Conference on Neural Networks*, 2:1339-1344, 1993.

S.S. Keerthi, S.K. Shevade, C. Bhattacharyya and K.R.K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation,* 13:637-649, 2001.

J.T.Y. Kwok. Moderating the outputs of support vector machine classifiers. *IEEE Transactions on Neural Networks*, 10(5):1018-1031, 1999.

T. Nayak and C.R. Rao. Cross entropy, dissimilarity measures and characterizations of quadratic entropy. *IEEE Trans. Information Theory*, IT-31:589-593, 1985.

N. Lawrence, M. Seeger and R. Herbrich. Fast sparse gaussian process methods: The informative vector machine. *Neural Information Processing Systems 15*, pages 609-616, 2003.

M. Oren, C. Papageorgiou, P. Sinha, E. Osuna and T. Poggio. Pedestrian detection using wavelet templates. *Computer Vision and Pattern Recognition (CVPR)*,pages 193-199, 1997.

E. Osuna, R. Freund and F. Girosi. Training support vector machines: An application to face detection. *Computer Vision and Pattern Recognition*, pages 130-136, 1997.

D.S. Pietra and D.V. Pietra. Statistical Modeling by ME. IBM Internal Report, 1993.

J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. Smola et al., Eds., *Adv. Large Margin Classifiers,* Cambridge MA: MIT Press, 1999.

J. Platt. Fast training of support vector machine using sequential minimal optimization. In Scholkopf, Burges and Smola, Eds., *Adv. Kernel Methods,* Cambridge MA: MIT Press, 1999.

M. Pontil and A. Verri. Properties of support vector machines. *Neural Computation*, 10:977-996, 1998.

M. Pontil and A. Verri. Support Vector Machines for 3-D Object Recognition. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 20:637-646, 1998.

L. Rabiner and B.H. Juang. *Fundamentals of Speech Recognition,* Englewood Cliffs, NJ: Prentice-Hall, 1993.

R.T. Rockefeller. *Convex Analysis*. Princeton Landmarks in Mathematics and Physics, Princeton University Press, 1970.

H.A. Rowley, S.A. Baluja and T. Kanade. Neural network based face detection. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 20(1):23-38, 1998.

M. Schmidt and H. Gish. Speaker identification via support vector classifiers. *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP'96),* 1:105-108, 1996.

B. Schölkopf, C. Burges and A. Smola Eds., *Adv. Kernel Methods-Support Vector Learning,* MIT Press, Cambridge MA, 1998.

B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond,* MIT Press, Cambridge MA, 2001.

M.E. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211-244, 2001.

V. Vapnik. *The Nature of Statistical Learning Theory.* New York: Springer-Verlag, 1995.

G. Wahba. Support vector machines, reproducing kernel Hilbert spaces and randomized GACV. *Adv. Kernel Methods-Support Vector Learning,* B. Schölkopf, C.J.C. Burges and A.J. Smola, Eds., Cambridge MA: MIT Press, 1998.

J. Weston and C. Watkins. Multi-class support vector machines. Technical Report CSD-TR-9800-04, Department of Computer Science, Royal Holloway, Univ. London, 1998.

J. Zhu and T. Hastie. Kernel logistic regression and import vector machine. *Adv. Neural Information Processing Systems (NIPS'2001)*, Cambridge, MA: MIT Press, 2002.

Journal of Machine Learning Research 8 (2007) 841-861 Submitted 11/06; Published 4/07

Preventing Over-Fitting during Model Selection via Bayesian Regularisation of the Hyper-Parameters

Gavin C. Cawley GCC@CMP.UEA.AC.UK
Nicola L. C. Talbot NLCT@CMP.UEA.AC.UK
School of Computing Sciences
University of East Anglia
Norwich, United Kingdom NR4 7TJ

Editors: Isabelle Guyon and Amir Saffari

Abstract

While the model parameters of a kernel machine are typically given by the solution of a convex optimisation problem, with a single global optimum, the selection of good values for the regularisation and kernel parameters is much less straightforward. Fortunately the leave-one-out cross-validation procedure can be performed or a least approximated very efficiently in closed form for a wide variety of kernel learning methods, providing a convenient means for model selection. Leave-one-out cross-validation based estimates of performance, however, generally exhibit a relatively high variance and are therefore prone to over-fitting. In this paper, we investigate the novel use of Bayesian regularisation at the second level of inference, adding a regularisation term to the model selection criterion corresponding to a prior over the hyper-parameter values, where the additional regularisation parameters are integrated out analytically. Results obtained on a suite of thirteen real-world and synthetic benchmark data sets clearly demonstrate the benefit of this approach.

Keywords: model selection, kernel methods, Bayesian regularisation

1. Introduction

Leave-one-out cross-validation (Lachenbruch and Mickey, 1968; Luntz and Brailovsky, 1969; Stone, 1974) provides the basis for computationally efficient model selection strategies for a variety of kernel learning methods, including the Support Vector Machine (SVM) (Cortes and Vapnik, 1995; Chapelle et al., 2002), Gaussian Process (GP) (Rasmussen and Williams, 2006; Sundararajan and Keerthi, 2001), Least-Squares Support Vector Machine (LS-SVM) (Suykens and Vandewalle, 1999; Cawley and Talbot, 2004), Kernel Fisher Discriminant (KFD) analysis (Mika et al., 1999; Cawley and Talbot, 2003; Saadi et al., 2004; Bo et al., 2006) and Kernel Logistic Regression (KLR) (Keerthi et al., 2005; Cawley and Talbot, 2007). These methods have proved highly successful for kernel machines having only a small number of hyper-parameters to optimise, as demonstrated by the set of models achieving the best average score in the WCCI-2006 performance prediction challenge[1] (Cawley, 2006; Guyon et al., 2006). Unfortunately, while leave-one-out cross-validation estimators have been shown to be almost unbiased (Luntz and Brailovsky, 1969), they are known to exhibit a relatively high variance (e.g., Kohavi, 1995). A kernel with many hyper-parameters, for instance those used in Automatic Relevance Determination (ARD) (e.g., Rasmussen and Williams, 2006) or feature scaling methods (Chapelle et al., 2002; Bo et al., 2006), may provide sufficient

1. See `http://www.modelselect.inf.ethz.ch/index.php`.

degrees of freedom to over-fit leave-one-out cross-validation based model selection criteria, resulting in performance inferior to that obtained using a less flexible kernel function. In this paper, we investigate the novel use of regularisation (Tikhonov and Arsenin, 1977) of the hyper-parameters in model selection in order to ameliorate the effects of the high variance of leave-one-out cross-validation based selection criteria, and so improve predictive performance. The regularisation term corresponds to a zero-mean Gaussian prior over the values of the kernel parameters, representing a preference for smooth kernel functions, and hence a relatively simple classifier. The regularisation parameters introduced in this step are integrated out analytically in the style of Buntine and Weigend (1991), to provide a Bayesian model selection criterion that can be optimised in a straightforward manner via, for example, scaled conjugate gradient descent (Williams, 1991).

The paper is structured as follows: The remainder of this section provides a brief overview of the least-squares support vector machine, including the use of leave-one-out cross-validation based model selection procedures, given in sufficient detail to ensure the reproducibility of the results. Section 2 describes the use of Bayesian regularisation to prevent over-fitting at the second level of inference, that is, model selection. Section 3 presents results obtained over a suite of thirteen benchmark data sets, which demonstrate the utility of this approach. Section 4 provides discussion of the results and suggests directions for further research. Finally, the work is summarised and directions for further work are outlined in Section 5.

1.1 Least Squares Support Vector Machine

In the remainder of this section, we provide a brief overview of the least-squares support vector machine (Suykens and Vandewalle, 1999) used as the testbed for the investigation of the role of regularisation in the model selection process described in this study. Given training data,

$$\mathcal{D} = \{(x_i,\ y_i)\}_{i=1}^{\ell}, \quad \text{where} \quad x_i \in \mathcal{X} \subset \mathbb{R}^d \quad \text{and} \quad y_i \in \{-1, +1\},$$

we seek to construct a linear discriminant, $f(x) = \phi(x) \cdot w + b$, in a *feature* space, $\mathcal{F}$, defined by a fixed transformation of the input space, $\phi : \mathcal{X} \to \mathcal{F}$. The parameters of the linear discriminant, $(w,\ b)$, are given by the minimiser of a *regularised* (Tikhonov and Arsenin, 1977) least-squares training criterion,

$$L = \frac{1}{2}\|w\|^2 + \frac{1}{2\mu}\sum_{i=1}^{\ell}\left[y_i - \phi(x_i)\cdot w - b\right]^2, \tag{1}$$

where μ is a regularisation parameter controlling the bias-variance trade-off (Geman et al., 1992). Rather than specify the feature space directly, it is instead induced by a kernel function, $\mathcal{K} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, which evaluates the inner-product between the projections of the data onto the feature space, $\mathcal{F}$, that is, $\mathcal{K}(x, x') = \phi(x)\cdot\phi(x')$. The interpretation of an inner-product in a fixed feature space is valid for any Mercer kernel (Mercer, 1909), for which the Gram matrix, $K = [k_{ij} = \mathcal{K}(x_i, x_j)]_{i,j=1}^{\ell}$ is positive semi-definite, that is,

$$a^T K a \geq 0, \qquad \forall\, a \in \mathbb{R}^{\ell}, \qquad a \neq 0.$$

The Gram matrix effectively encodes the spatial relationships between the projections of the data in the feature space, $\mathcal{F}$. A linear model can thus be implicitly constructed in the feature space using only information contained in the Gram matrix, without explicitly evaluating the positions of the data in the feature space via the transformation $\phi(\cdot)$. Indeed, the *representer* theorem (Kimeldorf and Wahba, 1971) shows that the solution of the optimisation problem (1) can be written as an expansion over the training patterns,

$$w = \sum_{i=1}^{\ell} \alpha_i \phi(x_i) \qquad \Longrightarrow \qquad f(x) = \sum_{i=1}^{\ell} \alpha_i \mathcal{K}(x_i, x) + b.$$

The advantage of the "kernel trick" then becomes apparent; a linear model can be constructed in an extremely rich, high- (possibly infinite-) dimensional feature space, using only finite-dimensional quantities, such as the Gram matrix, K. The "kernel trick" also allows the construction of statistical models that operate directly on structured data, for instance strings, trees and graphs, leading to the current interest in kernel learning methods in computational biology (Schölkopf et al., 2004) and text-processing (Joachims, 2002). The Radial Basis Function (RBF) kernel,

$$\mathcal{K}(x, x') = \exp\left\{-\eta \|x - x'\|^2\right\}$$

is commonly encountered in practical applications of kernel learning methods, here η is a *kernel parameter*, controlling the sensitivity of the kernel function. The feature space for the radial basis function kernel consists of the positive orthant of an infinite-dimensional unit hyper-sphere (e.g., Shawe-Taylor and Cristianini, 2004). The Gram matrix for the radial basis function kernel is thus of full rank (Micchelli, 1986), and so the kernel model is able to form an arbitrary shattering of the data.

1.1.1 A DUAL TRAINING ALGORITHM

The basic training algorithm for the least-squares support vector machine (Suykens and Vandewalle, 1999) views the regularised loss function (1) as a constrained minimisation problem:

$$\min_{w, b, \varepsilon_i} \frac{1}{2}\|w\|^2 + \frac{1}{2\mu}\sum_{i=1}^{\ell} \varepsilon_i^2 \qquad \text{subject to} \qquad \varepsilon_i = y_i - w \cdot \phi(x_i) - b.$$

The primal Lagrangian for this constrained optimisation problem gives the unconstrained minimisation problem defined by the following regularised loss function,

$$\mathcal{L} = \frac{1}{2}\|w\|^2 + \frac{1}{2\mu}\sum_{i=1}^{\ell} \varepsilon_i^2 - \sum_{i=1}^{\ell} \alpha_i \left\{w \cdot \phi(x_i) + b + \varepsilon_i - y_i\right\},$$

where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_\ell) \in \mathbb{R}^\ell$ is a vector of Lagrange multipliers. The optimality conditions for this problem can be expressed as follows:

$$\frac{\partial \mathcal{L}}{\partial w} = 0 \Longrightarrow w = \sum_{i=1}^{\ell} \alpha_i \phi(x_i), \tag{2}$$

$$\frac{\partial \mathcal{L}}{\partial b} = 0 \Longrightarrow \sum_{i=1}^{\ell} \alpha_i = 0, \tag{3}$$

$$\frac{\partial \mathcal{L}}{\partial \varepsilon_i} = 0 \Longrightarrow \alpha_i = \frac{\varepsilon_i}{\mu}, \quad \forall_i \in \{1, 2, \ldots, \ell\}, \tag{4}$$

$$\frac{\partial \mathcal{L}}{\partial \alpha_i} = 0 \Longrightarrow w \cdot \phi(x_i) + b + \varepsilon_i - y_i = 0, \quad \forall\, i \in \{1, 2, \ldots, \ell\}. \tag{5}$$

Using (2) and (4) to eliminate w and $\varepsilon = (\varepsilon_1, \varepsilon_2, \dots, \varepsilon_\ell)$, from (5), we find that

$$\sum_{j=1}^{\ell} \alpha_j \phi(x_j) \cdot \phi(x_i) + b + \mu\alpha_i = y_i \qquad \forall\, i \in \{1, 2, \dots, \ell\}. \tag{6}$$

Noting that $\mathcal{K}(x, x') = \phi(x) \cdot \phi(x')$, the system of linear equations, (6) and (3), can be written more concisely in matrix form as

$$\begin{bmatrix} K + \mu I & 1 \\ 1^T & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ b \end{bmatrix} = \begin{bmatrix} y \\ 0 \end{bmatrix},$$

where $K = [k_{ij} = \mathcal{K}(x_i, x_j)]_{i,j=1}^{\ell}$, I is the $\ell \times \ell$ identity matrix and 1 is a column vector of ℓ ones. The optimal parameters for the model of the conditional mean can then be obtained with a computational complexity of $O(\ell^3)$ operations, using direct methods, such as Cholesky decomposition (Golub and Van Loan, 1996).

1.1.2 EFFICIENT IMPLEMENTATION VIA CHOLESKY DECOMPOSITION

A more efficient training algorithm can be obtained, taking advantage of the special structure of the system of linear equations (Suykens et al., 2002). The system of linear equations to be solved in fitting a least-squares support vector machine is given by,

$$\begin{bmatrix} M & 1 \\ 1^T & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ b \end{bmatrix} = \begin{bmatrix} y \\ 0 \end{bmatrix}, \tag{7}$$

where $M = K + \mu I$. Unfortunately the matrix on the left-hand side is not positive definite, and so we cannot solve this system of linear equations directly using the Cholesky decomposition. However, the first row of (7) can be re-written as

$$M\left(\alpha + M^{-1}1b\right) = y. \tag{8}$$

Rearranging (8), we see that $\alpha = M^{-1}(y - 1b)$, using this result to eliminate α, the second row of (7) can be written as

$$1^T M^{-1} 1 b = 1^T M^{-1} y.$$

The system of linear equations can then be re-written as

$$\begin{bmatrix} M & 0 \\ 0^T & 1^T M^{-1} 1 \end{bmatrix} \begin{bmatrix} \alpha + M^{-1}1b \\ b \end{bmatrix} = \begin{bmatrix} y \\ 1^T M^{-1} y \end{bmatrix}. \tag{9}$$

In this case, the matrix on the left hand side is positive-definite, as $M = K + \lambda I$ is positive-definite and $1^T M^{-1} 1$ is positive since the inverse of a positive definite matrix is also positive definite. The revised system of linear equations (9) can be solved as follows: First solve

$$M\rho = 1 \qquad \text{and} \qquad M\nu = y, \tag{10}$$

which may be performed efficiently using the Cholesky factorisation of M. The model parameters of the least-squares support vector machine are then given by

$$b = \frac{1^T \nu}{1^T \rho} \qquad \text{and} \qquad \alpha = \nu - \rho b.$$

The two systems of linear equations (10) can be solved efficiently using the Cholesky decomposition of $M = R^T R$, where R is the upper triangular Cholesky factor of M.

1.2 Leave-One-Out Cross-Validation

Cross-validation (Stone, 1974) is commonly used to obtain a reliable estimate of the test error for performance estimation or for use as a model selection criterion. The most common form, k-fold cross-validation, partitions the available data into k disjoint subsets. In each iteration a classifier is trained on a different combination of $k-1$ subsets and the unused subset is used to estimate the test error rate. The k-fold cross-validation estimate of the test error rate is then simply the average of the test error rate observed in each of the k iterations, or folds. The most extreme form of cross-validation, where $k = \ell$ such that the test partition in each fold consists of only a single pattern, is known as leave-one-out cross-validation (Lachenbruch and Mickey, 1968) and has been shown to provide an almost unbiased estimate of the test error rate (Luntz and Brailovsky, 1969). Leave-one-out cross-validation is however computationally expensive, in the case of the least-squares support vector machine a naïve implementation having a complexity of $O(\ell^4)$ operations. Leave-one-out cross-validation is therefore normally only used in circumstances where the available data are extremely scarce such that the computational expense is no longer prohibitive. In this case the inherently high variance of the leave-one-out estimator (Kohavi, 1995) is offset by the minimal decrease in the size of the training set in each fold, and so may provide a more reliable estimate of generalisation performance than conventional k-fold cross-validation. Fortunately leave-one-out cross-validation of least-squares support vector machines can be performed in closed form with a computational complexity of only $O(\ell^3)$ operations (Cawley and Talbot, 2004). Leave-one-out cross-validation can then be used in medium to large scale applications, where there may be a few thousand data-points, although the relatively high variance of this estimator remains potentially problematic.

1.2.1 VIRTUAL LEAVE-ONE-OUT CROSS-VALIDATION

The optimal values of the parameters of a Least-Squares Support Vector Machine are given by the solution of a system of linear equations:

$$\begin{bmatrix} K+\mu I & 1 \\ 1^T & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ b \end{bmatrix} = \begin{bmatrix} y \\ 0 \end{bmatrix}. \tag{11}$$

The matrix on the left-hand side of (11) can be decomposed into block-matrix representation, as follows:

$$\begin{bmatrix} K+\mu I & 1 \\ 1^T & 0 \end{bmatrix} = \begin{bmatrix} c_{11} & c_1^T \\ c_1 & C_1 \end{bmatrix} = C.$$

Let $[\alpha^{(-i)}; b^{(-i)}]$ represent the parameters of the least-squares support vector machine during the i^{th} iteration of the leave-one-out cross-validation procedure, then in the first iteration, in which the first training pattern is excluded,

$$\begin{bmatrix} \alpha^{(-1)} \\ b^{(-1)} \end{bmatrix} = C_1^{-1}\left[y_2, \ldots, y_\ell, 0\right]^T.$$

The leave-one-out prediction for the first training pattern is then given by

$$\hat{y}_1^{(-1)} = c_1^T \begin{bmatrix} \alpha^{(-1)} \\ b^{(-1)} \end{bmatrix} = c_1^T C_1^{-1}\left[y_2, \ldots, y_\ell, 0\right]^T.$$

Considering the last ℓ equations in the system of linear equations (11), it is clear that $[c_1\ C_1]$ $[\alpha_1,\ldots,\alpha_\ell,b]^T = [y_2,\ldots,y_\ell,0]^T$, and so

$$\hat{y}_1^{(-1)} = c_1^T C_1^{-1}[c_1\ C_1]\left[\alpha^T,b\right]^T = c_1^T C_1^{-1} c_1 \alpha_1 + c_1[\alpha_2,\ldots,\alpha_\ell,b]^T.$$

Noting, from the first equation in the system of linear equations (11), that $y_1 = c_{11}\alpha_1 + c_1^T[\alpha_2,\ldots,\alpha_\ell,b]^T$, thus

$$\hat{y}_1^{(-1)} = y_1 - \alpha_1\left(c_{11} - c_1^T C_1^{-1} c_1\right).$$

Finally, via the block matrix inversion lemma,

$$\begin{bmatrix} c_{11} & c_1^T \\ c_1 & C_1 \end{bmatrix}^{-1} = \begin{bmatrix} \kappa^{-1} & -\kappa^{-1} c_1 C_1^{-1} \\ C_1^{-1} + \kappa^{-1} C_1^{-1} c_1^T c_1 C_1^{-1} & -\kappa^{-1} C_1^{-1} c_1^T \end{bmatrix},$$

where $\kappa = c_{11} - c_1^T C_1^{-1} c$, and noting that the system of linear equations (11) is insensitive to permutations of the ordering of the equations and of the unknowns, we have that,

$$y_i - \hat{y}_i^{(-i)} = \frac{\alpha_i}{C_{ii}^{-1}}. \tag{12}$$

This means that, assuming the system of linear equations (11) is solved via explicit inversion of C, a leave-one-out cross-validation estimate of an appropriate model selection criterion can be evaluated using information already available a by-product of training the least-squares support vector machine on the entire data set (cf., Sundararajan and Keerthi, 2001).

1.2.2 EFFICIENT IMPLEMENTATION VIA CHOLESKY FACTORISATION

The leave-one-out cross-validation behaviour of the least-squares support vector machine is described by (12). The coefficients of the kernel expansion, α, can be found efficiently, via Cholesky factorisation, as described in Section 1.1.2. However we must also determine the diagonal elements of C^{-1} in an efficient manner. Using the block matrix inversion formula, we obtain

$$C^{-1} = \begin{bmatrix} M & 1 \\ 1^T & 0 \end{bmatrix}^{-1} = \begin{bmatrix} M^{-1} + M^{-1} 1 S_M^{-1} 1^T M^{-1} & -M^{-1} 1 S_M^{-1} \\ -S_M^{-1} 1^T M^{-1} & S_M^{-1} \end{bmatrix}$$

where $M = K + \mu I$ and $S_M = -1^T M^{-1} 1 = -1^T \eta$ is the Schur complement of M. The inverse of the positive definite matrix, M, can be computed efficiently from its Cholesky factorisation, via the `SYMINV` algorithm (Seaks, 1972), for example using the LAPACK routine `DTRTRI` (Anderson et al., 1999). Let $R = [r_{ij}]_{i,j=1}^{\ell}$ be the lower triangular Cholesky factor of the positive definite matrix M, such that $M = RR^T$. Furthermore, let

$$S = [s_{ij}]_{i,j=1}^{\ell} = R^{-1}, \qquad \text{where} \qquad s_{ii} = \frac{1}{r_{ii}} \qquad \text{and} \qquad s_{ij} = -s_{ii}\sum_{k=1}^{i-1} r_{ik} s_{kj},$$

represent the (lower triangular) inverse of the Cholesky factor. The inverse of M is then given by $M^{-1} = S^T S$. In the case of efficient leave-one-out cross-validation of least-squares support vector machines, we are principally concerned only with the diagonal elements of M^{-1}, given by

$$M_{ii}^{-1} = \sum_{j=1}^{i} s_{ij}^2 \qquad \Longrightarrow \qquad C_{ii}^{-1} = \sum_{j=1}^{i} s_{ij}^2 + \frac{\rho_i^2}{S_M} \qquad \forall\, i \in \{1,2,\ldots,\ell\}.$$

The computational complexity of the basic training algorithm is $O(\ell^3)$ operations, being dominated by the evaluation of the Cholesky factor. However, the computational complexity of the analytic leave-one-out cross-validation procedure, when performed as a by-product of the training algorithm, is only $O(\ell)$ operations. The computational expense of the leave-one-out cross-validation procedure therefore becomes increasingly negligible as the training set becomes larger.

1.3 Model Selection

The virtual leave-one-out cross-validation procedure described in the previous section provides the basis for a simple automated model selection strategy for the least-squares support vector machine. Perhaps the most basic model selection criterion is provided by the Predicted REsidual Sum of Squares (PRESS) criterion (Allen, 1974), which is simply the leave-one-out estimate of the sum-of-squares error,

$$Q(\theta) = \frac{1}{2}\sum_{i=1}^{\ell}\left[y_i - \hat{y}_i^{(-i)}\right]^2.$$

A minimum of the model selection criterion is often found via a simple grid-search procedure in the majority of practical applications of kernel learning methods. However, this is rarely necessary and often highly inefficient as a grid-search spends a large amount of time investigating hyper-parameter values outside the neighbourhood of the global optimum. A more efficient approach uses the Nelder-Mead simplex algorithm (Nelder and Mead, 1965), as implemented by the `fminsearch` function of the MATLAB optimisation toolbox. An alternative easily implemented approach uses conjugate gradient methods, with the required gradient information estimated by the method of finite differences, and implemented by the `fminunc` function from the MATLAB optimisation toolbox. In this study however, we use scaled conjugate gradient descent (Williams, 1991), with the required gradient information evaluated analytically, as this is approximately twice as efficient.

1.3.1 PARTIAL DERIVATIVES OF THE PRESS MODEL SELECTION CRITERION

Let $\theta = \{\theta_1, \ldots, \theta_n\} = \{\lambda, \eta_1, \ldots, \eta_d\}$ represent the vector of hyper-parameters for a least-squares support vector machine, where $\eta_1, \ldots, \eta_d$ represent the kernel parameters. The PRESS statistic (Allen, 1974) can be written as

$$Q(\theta) = \frac{1}{2}\sum_{i=1}^{\ell}\left[r_i^{(-i)}\right]^2, \qquad \text{where} \qquad r_i^{(-i)} = y_i - \hat{y}_i^{(-i)} = \frac{\alpha_i}{C_{ii}^{-1}}.$$

Using the chain rule, the partial derivative of the PRESS statistic, with respect to an individual hyper-parameter, θ_j, is given by,

$$\frac{\partial Q(\theta)}{\partial \theta_j} = \sum_{i=1}^{\ell}\frac{\partial Q(\theta)}{\partial r_i^{(-i)}}\frac{\partial r_i^{(-i)}}{\partial \theta_j},$$

where

$$\frac{\partial Q(\theta)}{\partial r_i^{(-i)}} = r_i^{(-i)} = \frac{\alpha_i}{C_{ii}^{-1}} \qquad \text{and} \qquad \frac{\partial r_i^{(-i)}}{\partial \theta_j} = \frac{\partial \alpha_i}{\partial \theta_j}\frac{1}{C_{ii}^{-1}} - \frac{\alpha_i}{\left[C_{ii}^{-1}\right]^2}\frac{\partial C_{ii}^{-1}}{\partial \theta_j},$$

such that

$$\frac{\partial Q(\theta)}{\partial \theta_j} = \sum_{i=1}^{\ell}\frac{\alpha_i}{C_{ii}^{-1}}\left\{\frac{\partial \alpha_i}{\partial \theta_j}\frac{1}{C_{ii}^{-1}} - \frac{\alpha_i}{\left[C_{ii}^{-1}\right]^2}\frac{\partial C_{ii}^{-1}}{\partial \theta_j}\right\}.$$

We begin by deriving the partial derivatives of the model parameters, $\left[\alpha^T \; b\right]^T$, with respect to the hyper-parameter θ_j. The model parameters are given by the solution of a system of linear equations, such that

$$\left[\alpha^T \; b\right]^T = C^{-1}\left[y^T \; 0\right]^T.$$

Using the following identity for the partial derivatives of the inverse of a matrix,

$$\frac{\partial C^{-1}}{\partial \theta_j} = -C^{-1}\frac{\partial C}{\partial \theta_j}C^{-1}, \tag{13}$$

we obtain,

$$\frac{\partial \left[\alpha^T \; b\right]^T}{\partial \theta_j} = -C^{-1}\frac{\partial C}{\partial \theta_j}C^{-1}\left[y^T \; 0\right] = -C^{-1}\frac{\partial C}{\partial \theta_j}\left[\alpha^T \; b\right]^T.$$

Note the computational complexity of evaluating the partial derivatives of the model parameters is $O(\ell^2)$, as only two successive matrix-vector products are required. The partial derivatives of the diagonal elements of C^{-1} can be found using the inverse matrix derivative identity (13). For a kernel parameter, $\partial C/\partial \eta_j$ will generally be fully dense, and so the computational complexity of evaluating the diagonal elements of $\partial C^{-1}/\partial \eta_j$ will be $O(\ell^3)$ operations. If, on the other hand, we consider the regularisation parameter, μ, we have that

$$\frac{\partial C}{\partial \mu} = \begin{bmatrix} I & 0 \\ 0^T & 0 \end{bmatrix},$$

and so the computation of the partial derivatives of the model parameters, with respect to the regularisation parameter, is slightly simplified,

$$\frac{\partial \left[\alpha^T \; b\right]^T}{\partial \mu} = -C^{-1}\left[\alpha^T \; b\right]^T.$$

More importantly, as $\partial C/\partial \mu$ is diagonal, the diagonal elements of (13) can be evaluated with a computational complexity of only $O(\ell^2)$ operations. This suggests that it may be more efficient to adopt different strategies for optimising the regularisation parameter, μ, and the vector of kernel parameters, η, (cf., Saadi et al., 2004). For a kernel parameter, η_j, the partial derivatives of C with respect to η_j are given by the partial derivatives of the kernel matrix, that is,

$$\frac{\partial C}{\partial \eta_j} = \begin{bmatrix} \partial K/\partial \eta_j & 0 \\ 0^T & 0 \end{bmatrix}.$$

For the spherical radial basis function kernel, used in this study, the partial derivative with respect to the kernel parameter is given by

$$\frac{\partial \mathcal{K}(x, x')}{\partial \eta} = -\mathcal{K}(x, x')\|x - x'\|^2.$$

Finally, since the regularisation parameter, μ, and the scale parameter of the radial basis function kernel are strictly positive quantities, in order to permit the use of an unconstrained optimisation procedure, we adopt the parameterisation $\widetilde{\theta}_j = \log_2 \theta_j$, such that

$$\frac{\partial Q(\theta)}{\partial \widetilde{\theta}_j} = \frac{\partial Q(\theta)}{\partial \theta_j}\frac{\partial \theta_j}{\partial \widetilde{\theta}_j} \qquad \text{where} \qquad \frac{\partial \theta_j}{\partial \widetilde{\theta}_j} = \theta_j \log 2.$$

1.3.2 AUTOMATIC RELEVANCE DETERMINATION

Automatic Relevance Determination (ARD) (e.g., Rasmussen and Williams, 2006), also known as feature scaling (Chapelle et al., 2002; Bo et al., 2006), aims to identify informative input features as a natural consequence of optimising the model selection criterion. This can be most easily achieved using an *elliptical* radial basis function kernel,

$$\mathcal{K}(x,x') = \exp\left\{ -\sum_{i=1}^{d} \eta_i [x_i - x'_i]^2 \right\},$$

that incorporates individual scaling factors for each input dimension. The partial derivatives with respect to the kernel parameters are then given by,

$$\frac{\partial \mathcal{K}(x,x')}{\partial \eta_i} = -\mathcal{K}(x,x')[x_i - x'_i]^2.$$

Generalisation performance is likely to be enhanced if irrelevant features are down-weighted. It is therefore hoped that minimising the model selection criterion will lead to very small values for the scaling factors associated with redundant input features, allowing them to be identified and pruned from the model.

2. Bayesian Regularisation in Model Selection

In order to overcome the observed over-fitting in model selection using leave-one-out cross-validation based methods, we propose to add a regularisation term (Tikhonov and Arsenin, 1977) to the model selection criterion, which penalises solutions where the kernel parameters take on unduly large values. The regularised model selection criterion is then given by

$$M(\theta) = \zeta Q(\theta) + \xi \Omega(\theta), \tag{14}$$

where ξ and ζ are additional regularisation parameters, $Q(\theta)$ is the model selection criterion, in this case the PRESS statistic and $\Omega(\theta)$ is a regularisation term,

$$Q(\theta) = \frac{1}{2}\sum_{i=1}^{\ell}\left[y_i - \hat{y}_i^{(-i)}\right]^2 \qquad \text{and} \qquad \Omega(\theta) = \frac{1}{2}\sum_{i=1}^{d}\eta_i^2.$$

In this study we have left the regularisation parameter, μ, unregularised. However, we have now introduced two further regularisation parameters ξ and ζ for which good values must also be found. This problem may be solved by taking a Bayesian approach and adopting an ignorance prior and integrating out the additional regularisation parameters analytically in the style of Buntine and Weigend (1991). Adapting the approach taken by Williams (1995), the regularised model selection criterion (14) can be interpreted as the posterior density in the space of the hyper-parameters,

$$P(\theta|\mathcal{D}) \propto P(\mathcal{D}|\theta)P(\theta),$$

by taking the negative logarithm and neglecting additive constants. Here $P(\mathcal{D}|\theta)$ represents the *likelihood* with respect to the hyper-parameters and $P(\theta)$ represents our prior beliefs regarding the hyper-parameters, in this case that they should have a small magnitude, corresponding to a relatively simple model. These quantities can be expressed as

$$P(\mathcal{D}|\theta) = Z_Q^{-1} \exp\{-\zeta Q(\theta)\} \qquad \text{and} \qquad P(\theta) = Z_\Omega^{-1} \exp\{-\xi\Omega(\theta)\}$$

where Z_Q and Z_Ω are the appropriate normalising constants. Assuming the data represent an i.i.d. sample, the *likelihood* in this case is Gaussian,

$$P(\mathcal{D}|\theta) = \prod_{i=1}^{\ell} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{ -\frac{\left[y_i - \hat{y}_i^{(-i)}\right]^2}{2\sigma^2} \right\} \quad \text{where} \quad \zeta = \frac{1}{\sigma^2} \implies Z_Q = \left(\frac{2\pi}{\zeta}\right)^{\ell/2}.$$

Likewise, the *prior* is a Gaussian, centred on the origin,

$$P(\theta) = \prod_{i=1}^{d} \frac{1}{\sqrt{2\pi/\xi}} \exp\left\{ -\frac{\xi}{2}\eta_i^2 \right\} \qquad \text{such that} \qquad Z_\Omega = \left(\frac{2\pi}{\xi}\right)^{d/2}.$$

Minimising (14) is thus equivalent to maximising the posterior density with respect to the hyper-parameters. Note that the use of a prior over the hyper-parameters is in accordance with normal Bayesian practice and has been investigated in the case of Gaussian Process classifiers by Williams and Barber (1998). The combination of frequentist and Bayesian approaches at the first and second levels of inference is however somewhat unusual. The marginal likelihood is dependent on the assumptions of the model, which may not be completely appropriate. Cross-validation based procedures may therefore be more robust in the case of model mis-specification (Wahba, 1990). It seems reasonable for the model to be less sensitive to assumptions at the second level of inference than the first, and so the proposed approach represents a pragmatic combination of techniques.

2.1 Elimination of Second Level Regularisation Parameters ξ and ζ

Under the *evidence framework* proposed by MacKay (1992a,b,c) the hyper-parameters ξ and ζ are determined by maximising the marginal likelihood, also known as the Bayesian *evidence* for the model. In this study, however we opt to integrate out the hyper-parameters analytically, extending the work by Buntine and Weigend (1991) and Williams (1995) to consider Bayesian regularisation at the second level of inference, namely the selection of good values for the hyper-parameters. We begin with the prior over the hyper-parameters, which depends on ξ,

$$P(\theta|\xi) = Z_\Omega(\xi)^{-1} \exp\{-\xi\Omega\}.$$

The regularisation parameter ξ may then be integrated out analytically using a suitable prior, $P(\xi)$,

$$P(\theta) = \int P(\theta|\xi) P(\xi) d\xi.$$

The improper Jeffreys' prior, $P(\xi) \propto 1/\xi$ is an appropriate ignorance prior in this case as ξ is a scale parameter, noting that ξ is strictly positive,

$$p(\theta) = \frac{1}{(2\pi)^{d/2}} \int_0^\infty \xi^{d/2-1} \exp\{-\xi\Omega\} d\xi.$$

Using the Gamma integral $\int_0^\infty x^{\nu-1}e^{-\mu x}dx = \Gamma(\nu)/\mu^\nu$ (Gradshteyn and Ryzhic, 1994, equation 3.384), we obtain

$$p(\theta) = \frac{1}{(2\pi)^{d/2}}\frac{\Gamma(d/2)}{\Omega^{d/2}} \implies -\log p(\theta) \propto \frac{d}{2}\log\Omega.$$

Finally, adopting a similar procedure to eliminate ζ, we obtain a revised model selection criterion with Bayesian regularisation,

$$L = \frac{\ell}{2}\log Q(\theta) + \frac{d}{2}\log\Omega(\theta). \tag{15}$$

in which the regularisation parameters have been eliminated. As before, this criterion can be optimised via standard methods, such as the Nelder-Mead simplex algorithm (Nelder and Mead, 1965) or scaled conjugate gradient descent (Williams, 1991). The partial derivatives of the proposed Bayesian model selection criterion are given by

$$\frac{\partial L}{\partial\theta_i} = \frac{\ell}{2Q(\theta)}\frac{\partial Q(\theta)}{\partial\theta_i} + \frac{d}{2\Omega(\theta)}\frac{\partial\Omega(\theta)}{\partial\theta_i} \qquad \text{and} \qquad \frac{\partial\Omega(\theta)}{\partial\eta_i} = \eta_i.$$

The additional computational expense involved in Bayesian regularisation of the model selection criterion is only $O(d)$ operations, and is extremely small in comparison with the $O(\ell^3)$ operations involved in obtaining the leave-one-out error (including the cost of training the model on the entire data set). Per iteration of the model selection process, the cost of the Bayesian regularisation is therefore minimal. There seems little reason to suppose that the regularisation will have an adverse effect on convergence, and this seems to be the case in practice.

2.2 Relationship with the Evidence Framework

Under the evidence framework of MacKay (1992a,b,c) the regularisation parameters, ξ and ζ, are selected so as to maximise the marginal likelihood, also known as the Bayesian evidence, for the model. The log-evidence is given by

$$\log P(\mathcal{D}) = -\xi\Omega(\theta) - \zeta Q(\theta) - \frac{1}{2}\log|A| + \frac{d}{2}\log\xi + \frac{\ell}{2}\log\zeta - \frac{\ell}{2}\log\{2\pi\},$$

where A is the Hessian of the regularised model selection criterion (14) with respect to the hyper-parameters, θ. Setting the partial derivatives of the log evidence with respect to the regularisation parameters, ξ and ζ, equal to zero, we obtain the familiar update formulae,

$$\xi^{\text{new}} = \frac{\gamma}{2\Omega(\theta)} \qquad \text{and} \qquad \zeta^{\text{new}} = \frac{\ell-\gamma}{2Q(\theta)},$$

where γ is the number of well defined hyper-parameters, that is, the hyper-parameters for which the optimal value is primarily determined by the log-likelihood term, $Q(\theta)$ rather than by the regulariser, $\Omega(\theta)$. In the case of the L2 regularisation term, corresponding to a Gaussian prior, the number of well determined hyper-parameters is given by

$$\gamma = \sum_{j=1}^{n}\frac{\lambda_j}{\lambda_j + \xi}$$

where, $\lambda_i, \ldots, \lambda_n$ represent the eigenvalues of the Hessian of the unregularised model selection criterion, $Q(\theta)$ with respect to the kernel parameters. Comparing the partial derivatives of the regularised model selection criterion (14) with those of the Bayesian criterion (15), reveals that the

Bayesian regularisation scheme is equivalent to optimising the regularised model selection criterion (14) assuming that the regularisation parameters, ξ and ζ, are continuously updated according to the following update rules,

$$\xi^{\text{eff}} = \frac{d}{2\Omega(\theta)} \qquad \text{and} \qquad \zeta^{\text{eff}} = \frac{\ell}{2Q(\theta)}.$$

This exactly corresponds to the "cheap and cheerful" approximation of the evidence framework suggested by MacKay (1994), which assumes that all of the hyper-parameters are well-determined and that the number of hyper-parameters is small in relation to the number of training patterns. Since $\gamma \leq d$, it seems self evident that the proposed Bayesian regularisation scheme will be prone to a degree of under-fitting, especially in the case of a feature scaling kernel with many redundant features. The theoretical and practical pros and cons of the integrate-out approach and the evidence framework are discussed in some detail by MacKay (1994) and Bishop (1995) and references therein. However, the integrate-out approach does not require the evaluation of the Hessian matrix of the original selection criterion, $Q(\theta)$, which is likely to prove computationally prohibitive.

3. Results

In this section, we present experimental results demonstrating the benefits of the proposed model selection strategy incorporating Bayesian regularisation to overcome the inherent high variance of leave-one-out cross-validation based selection criteria. Table 2 shows a comparison of the error rates of least-squares support vector machines, using model selection procedures with, and without, Bayesian regularisation, (LS-SVM and LS-SVM-BR respectively) over the suite of thirteen public domain benchmark data sets used in the study by Mika et al. (2000). Results obtained using a Gaussian process classifier (Rasmussen and Williams, 2006), based on the expectation propagation method, are also provided for comparison (EP-GPC). The same set of 100 random partitions of the data (20 in the case of the larger `image` and `splice` benchmarks) to form training and test sets used in that study are also used here. In each case, model selection is performed independently for each realisation of the data set, such that the standard errors reflect the variability of both the training algorithm and the model selection procedure with changes in the sampling of the data. Both conventional spherical and elliptical radial basis kernels are used for all kernel learning methods, so that the effectiveness of each algorithm for automatic relevance determination can be assessed. The use of multiple training/test partitions allows an estimate of the statistical significance of differences in performance between algorithms to be computed. Let $\hat{x}$ and $\hat{y}$ represent the means of the performance statistic for a pair of competing algorithms, and e_x and e_y the corresponding standard errors, then the z statistic is computed as

$$z = \frac{\hat{y} - \hat{x}}{\sqrt{e_x^2 + e_y^2}}.$$

The z-score can then be converted to a significance level via the normal cumulative distribution function, such that $z = 1.64$ corresponds to a 95% significance level. All statements of statistical significance in the remainder of this section refer to a 95% level of significance.

3.1 Performance of Models Based on the Spherical RBF Kernel

The results shown in the first three data columns of Table 2 show the performance of LS-SVM, LS-SVM-BR and EP-ARD models based on the spherical Gaussian kernel. The performance of

Data Set	Training Patterns	Testing Patterns	Number of Replications	Input Features
Banana	400	4900	100	2
Breast cancer	200	77	100	9
Diabetis	468	300	100	8
Flare solar	666	400	100	9
German	700	300	100	20
Heart	170	100	100	13
Image	1300	1010	20	18
Ringnorm	400	7000	100	20
Splice	1000	2175	20	60
Thyroid	140	75	100	5
Titanic	150	2051	100	3
Twonorm	400	7000	100	20
Waveform	400	4600	100	21

Table 1: Details of data sets used in empirical comparison.

LS-SVM models with and without Bayesian regularisation are very similar, with neither model proving significantly better than the other on any of the data sets. This seems reasonable given that only two hyper-parameters are optimised during model selection and so there is little scope for over-fitting the PRESS model selection criterion and the regularisation term will have little effect. The LS-SVM model with Bayesian regularisation is significantly out-performed by the Gaussian Process classifier on one benchmark `banana`, but performs significantly better on a further four (`ringnorm`, `splice`, `twonorm`, `waveform`). Demšar (2006) recommends the use of the Wilcoxon signed rank test for assessing the statistical significance of differences in performance over multiple data sets. According to this test, neither the LSSVM-BR nor the EP-PGC is statistically superior at the 95% level of significance.

3.2 Performance of Models Based on the Elliptical RBF Kernel

The performances of LS-SVM, LS-SVM-BR and EP-GPC models based on the elliptical Gaussian kernel, which includes a separate scale parameter for each input feature, are shown in the last three columns of Table 2. Before evaluating the effects of Bayesian regularisation in model selection it is worth noting that the use of elliptical RBF kernels does not generally improve performance. For the LS-SVM, the elliptical kernel produces significantly better results on only two benchmarks (`image` and `splice`) and significantly worse results on a further eight (`banana`, `breast cancer`, `diabetis`, `german`, `heart`, `ringnorm`, `twonorm`, `waveform`), with the degradation in performance being very large in some cases (e.g., `heart`). This seems likely to be a result of the additional degrees of freedom involved in the model selection process, allowing over-fitting of the PRESS model selection criterion as a result of its inherently high variance. Note that fully Bayesian approaches, such as the Gaussian Process Classifier, are also unable to reliably select kernel parameters for the elliptical RBF kernel. The elliptical kernel is significantly better on only three benchmarks (`flare solar`,

Data Set	**Radial Basis Function**			**Automatic Relevance Determination**		
	LSSVM	**LSSVM-BR**	**EP-GPC**	**LSSVM**	**LSSVM-BR**	**EP-GPC**
Banana	10.60 ± 0.052	10.59 ± 0.050	**10.41 ± 0.046**	10.79 ± 0.072	10.73 ± 0.070	*10.46 ± 0.049*
Breast cancer	*26.73 ± 0.466*	27.08 ± 0.494	**26.52 ± 0.489**	29.08 ± 0.415	27.81 ± 0.432	27.97 ± 0.493
Diabetes	23.34 ± 0.166	**23.14 ± 0.166**	*23.28 ± 0.182*	24.35 ± 0.194	23.42 ± 0.177	23.86 ± 0.193
Flare solar	34.22 ± 0.169	34.07 ± 0.171	34.20 ± 0.175	34.39 ± 0.194	*33.61 ± 0.151*	**33.58 ± 0.182**
German	*23.55 ± 0.216*	23.59 ± 0.216	**23.36 ± 0.211**	26.10 ± 0.261	23.88 ± 0.217	23.77 ± 0.221
Heart	*16.64 ± 0.358*	**16.19 ± 0.348**	16.65 ± 0.287	23.65 ± 0.355	17.68 ± 0.623	19.68 ± 0.366
Image	3.00 ± 0.158	2.90 ± 0.154	2.80 ± 0.123	**1.96 ± 0.115**	*2.00 ± 0.113*	2.16 ± 0.068
Ringnorm	**1.61 ± 0.015**	**1.61 ± 0.015**	*4.41 ± 0.064*	2.11 ± 0.040	1.98 ± 0.026	8.58 ± 0.096
Splice	10.97 ± 0.158	10.91 ± 0.154	11.61 ± 0.181	*5.86 ± 0.179*	**5.14 ± 0.145**	7.07 ± 0.765
Thyroid	4.68 ± 0.232	4.63 ± 0.218	*4.36 ± 0.217*	4.68 ± 0.199	4.71 ± 0.214	**4.24 ± 0.218**
Titanic	**22.47 ± 0.085**	22.59 ± 0.120	22.64 ± 0.134	*22.58 ± 0.108*	22.86 ± 0.199	22.73 ± 0.134
Twonorm	**2.84 ± 0.021**	**2.84 ± 0.021**	*3.06 ± 0.034*	5.18 ± 0.072	4.53 ± 0.077	4.02 ± 0.068
Waveform	*9.79 ± 0.045*	**9.78 ± 0.044**	10.10 ± 0.047	13.56 ± 0.141	11.48 ± 0.177	11.34 ± 0.195

Table 2: Error rates of least-squares support vector machine, with and without Bayesian regularisation of the model selection criterion, in this case the PRESS statistic (Allen, 1974), and Gaussian process classifiers over thirteen benchmark data sets (Rätsch et al., 2001), using both standard radial basis function and automatic relevance determination kernels. The results for the EP-GPC were obtained using the MATLAB software accompanying the book by Rasmussen and Williams (2006). The results for each method are presented in the form of the mean error rate over test data for 100 realisations of each data set (20 in the case of the image and splice data sets), along with the associated standard error. The best results are shown in boldface and the second best in italics (without implication of statistical significance).

image and splice), while being significantly worse on six (breast cancer, diabetis, heart, ringnorm, twonorm and waveform).

In the case of the elliptical RBF kernel, the use of Bayesian regularisation in model selection has a dramatic effect on the performance of LS-SVM models, with the LS-SVM-BR model proving significantly better than the conventional LS-SVM on nine of the thirteen benchmarks (breast cancer, diabetis, flare solar, german, heart, ringnorm, splice, twonorm and waveform) without being significantly worse on any of the remaining four data sets. This demonstrates that over-fitting in model selection, due to the larger number of kernel parameters, is likely to be the significant factor causing the relatively poor performance of models with the elliptical RBF kernel. Again, the Gaussian Process classifier is significantly better than the LS-SVM with Bayesian regularisation on the banana and twonorm data sets, but is significantly worse on four of the remaining eleven (diabetis, heart, ringnorm and splice). Again, according to the Wilcoxon signed rank test, neither the LSSVM-BR nor the EP-PGC is statistically superior at the 95% level of significance. However the magnitude of the difference in performance between LSSVM-BR and EP-GPC approaches tends to be greatest when the LSSVM-BR out-performs EP-GPC, most notably on the heart, splice and ringnorm data sets. This provides some support for the observation of Wahba (1990) that cross-validation based model selection procedures should be more robust against model mis-specification (see also Rasmussen and Williams, 2006).

4. Discussion

The experimental evaluation presented in the previous section demonstrates that over-fitting can occur in model selection, due to the variance of the model selection criterion. In many cases the minimum of the selection criterion using the elliptical RBF kernel is lower than that achievable using the spherical RBF kernel, however this results in a degradation in generalisation performance. Using the PRESS statistic, the over-fitting is likely to be most severe in cases with a small number of training patterns, as the variance of the leave-one-out estimator decreases as the sample size becomes larger. Using the standard LSSVM, the elliptical RBF kernel only out-performs the spherical RBF kernel on two of the thirteen data sets, image and splice, which also happen to be the two largest data sets in terms of the number of training patterns. The greatest degradation in performance is obtained on the heart benchmark, the third smallest. The heart data set also has a relatively large number of input features (13). A large number of input features introduces a many additional degrees of freedom to optimise the model selection criterion, and so will generally tend to encourage over-fitting. However, there may be a compact subset of highly relevant features with the remainder being almost entirely uninformative. In this case the advantage of suppressing the noisy inputs is so great that it overcomes the predisposition towards over-fitting, and so results in improved generalisation (as observed in the case of the image and splice benchmarks). Whether the use of an elliptical RBF kernel will improve or degrade generalisation largely depends on such characteristics of the data that are not known *a-priori*, and so it seems prudent to consider a range of kernel functions and select the best via cross-validation.

The experimental results indicate that Bayesian regularisation of the hyper-parameters is generally beneficial, without at this stage providing a complete solution to the problem of over-fitting the model selection criterion. The effectiveness of the Bayesian regularisation scheme is to a large extent dependent on the appropriateness of the prior imposed on the hyper-parameters. There is no reason to assume that the simple Gaussian prior used here is in any sense optimal, and this is an issue where further research is necessary (see Section 4.2). The comparison of the integrate-out approach and the evidence framework highlights a deficiency of the simple Gaussian prior. It suggests that the integrate-out approach is likely to result in mild over-regularisation of the hyper-parameters in the presence of a large number of irrelevant features, as the corresponding hyper-parameters will be ill-determined.

The LSSVM with Bayesian regularisation of the hyper-parameters does not significantly out-perform the expectation propagation based Gaussian process classifier over the suite of thirteen benchmark data sets considered. This is not wholly surprising as the EP-GPC is at least very close to the state-of-the-art, indeed it is interesting that the EP-GPC does not out-perform such a comparatively simple model. The EP-GPC uses the marginal likelihood as the model selection criterion, which gives the probability of the data, *given the assumptions of the model* (Rasmussen and Williams, 2006). Cross-validation based approaches, on the other hand, provide an estimate of generalisation performance that does not depend on the model assumptions, and so may be more robust against model mis-specification (Wahba, 1990). The no free lunch theorems suggest that, at least in terms of generalisation performance, there is a lack of inherent superiority of one classification method over another, in the absence of *a-priori* assumptions regarding the data. This implies that if one classifier performs better than another on a particular data set it is because the inductive biases of that classifier provide a better fit to the particular pattern recognition task, rather than to its superiority in a more general sense. A model with strong inductive biases is likely to benefit when these biases are well suited to the data, but will perform badly when they do not. While a model with weak inductive biases will be more robust, it is less likely to perform conspicuously well on any given data set. This means there are complementary advantages and disadvantages to both approaches.

4.1 Relationship to Existing Work

The use of a prior over the hyper-parameters is in accordance with normal Bayesian practice and has been used in Gaussian Process classification (Williams and Barber, 1998). The problem of over-fitting in model selection has also been addressed by Qi et al. (2004), in the case of selecting informative features for a logistic regression model using an Automatic Relevance Determination (ARD) prior (cf., Tipping, 2000). In this case, the Expectation Propagation method (Minka, 2001) is used to obtain a deterministic approximation of the posterior, and also (as a by-product) a leave-one-out performance estimate. The latter is then used to implement a form of early-stopping (e.g., Sarle, 1995) to prevent over-fitting resulting from the direct optimization of the marginal likelihood until convergence. It seems likely that this approach would be also be beneficial in the case of tuning the hyper-parameters of the covariance function of Gaussian process model, using either the leave-one-out estimate arising from the EP approximation, or an approximate leave-one-out estimate from the Laplace approximation (cf., Cawley and Talbot, 2007).

4.2 Directions for Further Research

In this paper, the regularisation term corresponds to a simple spherical Gaussian prior over the kernel parameters. One direction of research would be to investigate alternative regularisation terms. The first possibility would be to use a regularisation term corresponding to a separable Laplace prior,

$$\Omega(\theta) = \frac{1}{2}\sum_{i=1}^{d} |\eta_i| \qquad \Longrightarrow \qquad p(\theta) = \prod_{i=1}^{d} \frac{\xi}{2}\exp\{-\xi|\eta_i|\}.$$

Setting the derivative of the regularised model selection criterion (14) to zero, we obtain

$$\left|\frac{\partial Q}{\partial \eta_i}\right| = \frac{\xi}{\zeta} \quad \text{if } |\eta_i| > 0 \qquad \text{and} \qquad \left|\frac{\partial Q}{\partial \eta_i}\right| < \frac{\xi}{\zeta} \quad \text{if } |\eta_i| = 0,$$

which implies that if the sensitivity of the leave-one-out error, $Q(\theta)$, falls below ξ/ζ, the value of the hyper-parameter, η_i will be set exactly to zero, effectively pruning that input from the model. In this way explicit feature selection may be obtained as a consequence of (regularised) model selection. The model selection criterion with Bayesian regularisation then becomes

$$L = \frac{\ell}{2}\log Q(\theta) + N \log \Omega(\theta)$$

where N is the number of input features with non-zero scale factors. This potentially overcomes the propensity towards under-fitting the data that might be expected when using the Gaussian prior, as the pruning action of the Laplace prior means that the values of all remaining hyper-parameters are well-determined by the data. In the case of the Laplace prior, the integrate-out approach is exactly equivalent to continuous updates of the hyper-parameters according to the update formulae under the evidence framework (Williams, 1995). Alternatively, defining a prior over the *function* of a model seems more in accordance with Bayesian ideals than choosing a prior over the parameters of the model. The use of a prior over the hyper-parameters based on the smoothness of the resulting model also provides a potential direction for future research. In this case, the regularisation term might take the form,

$$\Omega(\theta) = \frac{1}{2\ell}\sum_{i=1}^{\ell}\sum_{j=1}^{d}\left[\frac{\partial^2 \hat{y}_i}{\partial x_{ij}^2}\right]^2,$$

directly penalising models with excess curvature. This regularisation term corresponds to curvature driven smoothing in multi-layer perceptron networks (Bishop, 1993), except that the model output $\hat{y}_i$ is viewed as a function of the hyper-parameters, rather than of the weights. A penalty term based on the first-order partial derivatives is also feasible (cf., Drucker and Le Cun, 1992).

5. Conclusion

Leave-one-out cross-validation has proved to be an effective means of model selection for a variety of kernel learning methods, provided the number of hyper-parameters to be tuned is relatively small. The use of kernel functions with large numbers of parameters often provides sufficient degrees of freedom to over-fit the model selection criterion, leading to poor generalisation. In this paper, we have proposed the use of regularisation at the second level of inference, that is, model selection. The use of Bayesian regularisation is shown to be effective in reducing over-fitting, by ensuring the values of the kernel parameters remain small, giving a smoother kernel and hence a less complex classifier. This is achieved with only a minimal computational expense as the additional regularisation parameters are integrated out analytically using a reference prior. While a fully Bayesian model selection strategy is conceptually more elegant, it may also be less robust to model mis-specification. The use of leave-one-out cross-validation in model selection and Bayesian methods at the next level seems to be a pragmatic compromise. The effectiveness of this approach is clearly demonstrated in the experimental evaluation where, on average, the LS-SVM with Bayesian regularisation out-performs the expectation-propagation based Gaussian process classifier, using both spherical and elliptical RBF kernels.

Acknowledgments

The authors would like to thank the organisers of the WCCI model selection workshop and performance prediction challenge and the NIPS multi-level inference workshop and model selection game, and fellow participants for the stimulating discussions that have helped to shape this work. We also thank Carl Rasmussen and Chris Williams for their advice regarding the EP-GPC and the anonymous reviewers for their detailed and constructive comments that have significantly improved this paper.

References

D. M. Allen. The relationship between variable selection and prediction. *Technometrics*, 16:125–127, 1974.

E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorenson. *LAPACK Users' Guide*. SIAM Press, third edition, 1999.

C. M. Bishop. Curvature-driven smoothing: a learning algorithm for feedforward networks. *IEEE Transactions on Neural Networks*, 4(5):882–884, September 1993.

C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

L. Bo, L. Wang, and L. Jiao. Feature scaling for kernel Fisher discriminant analysis using leave-one-out cross-validation. *Neural Computation*, 18:961–978, April 2006.

W. L. Buntine and A. S. Weigend. Bayesian back-propagation. *Complex Systems*, 5:603–643, 1991.

G. C. Cawley. Leave-one-out cross-validation based model selection criteria for weighted LS-SVMs. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN-2006)*, pages 2970–2977, Vancouver, BC, Canada, July 16–21 2006.

G. C. Cawley and N. L. C. Talbot. Efficient leave-one-out cross-validation of kernel Fisher discriminant classifiers. *Pattern Recognition*, 36(11):2585–2592, November 2003.

G. C. Cawley and N. L. C. Talbot. Fast leave-one-out cross-validation of sparse least-squares support vector machines. *Neural Networks*, 17(10):1467–1475, December 2004.

G. C. Cawley and N. L. C. Talbot. Approximate leave-one-out cross-validation for kernel logistic regression. *Machine Learning* (submitted), 2007.

C. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1):131–159, 2002.

C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

H. Drucker and Y. Le Cun. Improving generalization performance using double back-propagation. *IEEE Transactions on Neural Networks*, 3(6):991–997, 1992.

S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilema. *Neural Computation*, 4(1):1–58, 1992.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, third edition edition, 1996.

I. S. Gradshteyn and I. M. Ryzhic. *Table of Integrals, Series and Products*. Academic Press, fifth edition, 1994.

I. Guyon, A. R. Saffari Azar Alamdari, G. Dror, and J. Buhmann. Performance prediction challenge. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN-2006)*, pages 1649–1656, Vancouver, BC, Canada, July 16–21 2006.

T. Joachims. *Learning to Classify Text using Support Vector Machines - Methods, Theory and Algorithms*. Kluwer Academic Publishers, 2002.

S. S. Keerthi, K. B. Duan, S. K. Shevade, and A. N. Poo. A fast dual algorithm for kernel logistic regression. *Machine Learning*, 61(1–3):151–165, November 2005.

G. S. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *J. Math. Anal. Applic.*, 33:82–95, 1971.

R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence (IJCAI)*, pages 1137–1143, San Mateo, CA, 1995. Morgan Kaufmann.

P. A. Lachenbruch and M. R. Mickey. Estimation of error rates in discriminant analysis. *Technometrics*, 10(1):1–11, February 1968.

A. Luntz and V. Brailovsky. On estimation of characters obtained in statistical procedure of recognition (in Russian). *Techicheskaya Kibernetica*, 3, 1969.

D. J. C. MacKay. Bayesian interpolation. *Neural Computation*, 4(3):415–447, 1992a.

D. J. C. MacKay. A practical Bayesian framework for backprop networks. *Neural Computation*, 4 (3):448–472, 1992b.

D. J. C. MacKay. The evidence framework applied to classification networks. *Neural Computation*, 4(5):720–736, 1992c.

D. J. C. MacKay. Hyperparameters: Optimise or integrate out? In G. Heidbreder, editor, *Maximum Entropy and Bayesian Methods*. Kluwer, 1994.

J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London, A*, 209:415–446, 1909.

C. A. Micchelli. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constructive Approximation*, 2:11–22, 1986.

S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing*, volume IX, pages 41–48. IEEE Press, New York, 1999.

S. Mika, G. Rätsch, J. Weston, B. Schölkopf, A. J. Smola, and K.-R. Müller. Invariant feature extraction and classification in feature spaces. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12, pages 526–532. MIT Press, 2000.

T. P. Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence*, pages 362–369. Morgan Kauffmann, 2001.

J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7: 308–313, 1965.

Y. Qi, T. P. Minka, R. W. Picard, and Z. Ghahramani. Predictive automatic relevance determination by expectation propagation. In *Proceedings of the 21st International Conference on Machine Learning*, pages 671–678, Banff, Alberta, Canada, July 4–8 2004.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, 2006.

G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3): 287–320, 2001.

K. Saadi, N. L. C. Talbot, and G. C. Cawley. Optimally regularised kernel Fisher discriminant analysis. In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR-2004)*, volume 2, pages 427–430, Cambridge, United Kingdom, August 23–26 2004.

W. S. Sarle. Stopped training and other remidies for overfitting. In *Proceedings of the 27*th *Symposium on the Interface of Computer Science and Statistics*, pages 352–360, Pittsburgh, PA, USA, June 21–24 1995.

B. Schölkopf, K. Tsuda, and J.-P. Vert. *Kernel Methods in Computational Biology*. MIT Press, 2004.

T. Seaks. SYMINV : An algorithm for the inversion of a positive definite matrix by the Cholesky decomposition. *Econometrica*, 40(5):961–962, September 1972.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society*, B 36(1):111–147, 1974.

S. Sundararajan and S. S. Keerthi. Predictive approaches for choosing hyperparameters in Gaussian processes. *Neural Computation*, 13(5):1103–1118, May 2001.

J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, June 1999.

J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, 2002.

A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-posed Problems*. John Wiley, New York, 1977.

M. E. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, June 2000.

G. Wahba. *Spline Models for Observational Data*. SIAM Press, Philadelphia, PA, 1990.

C. K. I. Williams and D. Barber. Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351, December 1998.

P. M. Williams. A Marquardt algorithm for choosing the step size in backpropagation learning with conjugate gradients. Technical Report CSRP-229, University of Sussex, February 1991.

P. M. Williams. Bayesian regularization and pruning using a Laplace prior. *Neural Computation*, 7 (1):117–143, 1995.

Journal of Machine Learning Research 8 (2007) 863-889 Submitted 6/05; Revised 10/06; Published 4/07

Combining PAC-Bayesian and Generic Chaining Bounds

Jean-Yves Audibert[*] AUDIBERT@CERMICS.ENPC.FR
CERTIS, Ecole Nationale des Ponts et Chausses
19, rue Alfred Nobel - Cité Descartes
F-77455 Marne-la-Vallée cedex 2, France

Olivier Bousquet[†] O.BOUSQUET@PERTINENCE.COM
Pertinence
32, rue des Jeûneurs
F-75002 Paris, France

Editor: John Shawe-Taylor

Abstract

There exist many different generalization error bounds in statistical learning theory. Each of these bounds contains an improvement over the others for certain situations or algorithms. Our goal is, first, to underline the links between these bounds, and second, to combine the different improvements into a single bound. In particular we combine the PAC-Bayes approach introduced by McAllester (1998), which is interesting for randomized predictions, with the optimal union bound provided by the generic chaining technique developed by Fernique and Talagrand (see Talagrand, 1996), in a way that also takes into account the variance of the combined functions. We also show how this connects to Rademacher based bounds.

Keywords: statistical learning theory, PAC-Bayes theorems, generalization error bounds

1. Introduction

Since the first results of Vapnik and Chervonenkis on uniform laws of large numbers for classes of $\{0;1\}$-valued functions, there has been a considerable amount of work aiming at obtaining generalizations and refinements of these bounds. This work has been carried out by different communities. On the one hand, people developing empirical processes theory like Dudley and Talagrand (among others) obtained very interesting results concerning the behavior of the suprema of empirical processes. On the other hand, people exploring learning theory tried to obtain refinements for specific algorithms with an emphasis on data-dependent bounds.

The goal of a generalization error bound is to control the behavior of the function that is returned by the algorithm. This function is data-dependent and thus unknown before seeing the data. As a consequence, if one wants to make statements about its error, one has to be able to *predict* which function is likely to be chosen by the algorithm. Since this cannot be done exactly, there is a need to provide guarantees that hold simultaneously for several candidate functions. This is known as the union bound. The way to perform this union bound optimally is now well mastered in the empirical

∗. Part of this work was done while this author was at laboratoire de Probabilités et Modèles Aléatoires, Université Paris 6, Paris

†. Part of this work was done while this author was at the Max Planck Institute for Biological Cybernetics, Tübingen

processes community. In particular, the role of the metric structure of the space of functions in the deviations of the empirical process has been thoroughly studied (see, for example, Talagrand, 2005).

In the learning theory setting, one is interested in bounds that are as algorithm and data dependent as possible. This particular focus has made concentration inequalities (see, for example, Boucheron et al., 2000) popular as they allow to obtain data-dependent results in an effortless way. Another aspect that is of interest for learning is the case where the classifiers are randomized or averaged. McAllester (1998, 1999) has proposed a new type of bound that takes the randomization into account in a clever way.

Another direction in which the error bounds can be improved is by using the variance of the functions in order to control the fluctuations of the empirical error. This idea originated in Huber's peeling device (here peeling refers to the fact that the class of function is "peeled off" into layers according to the variance of the functions) and is often referred to as "localization" (see, for example, van der Vaart and Wellner, 1996; van de Geer, 2000; Massart, 2000). It allows to get bounds with optimal rates of convergence, for example in the case of empirical error minimization (see, for example, Bartlett et al., 2005).

Our goal is to combine several of these improvements, bringing together the power of the majorizing measures as an optimal union bound technique and the power of the PAC-Bayesian bounds to handle randomized predictions efficiently in a way that is sensitive to the variance (localization) effect.

The paper is structured as follows. Next section introduces the notation and gives an overview of the existing bounds. Section 3 then presents our main result while Section 4 discusses its applications, showing in particular how to recover previously known results.

2. A Survey of Previous Results

In this section, after having introduced some notation and the setup of generalization bounds, we state and compare existing generalization bounds. By doing so, we hope to give a better and more global view on the various approaches that have been developed to obtain error bounds. In particular, we want to emphasize the differences and the complementarity of the approaches. Our goal is not to entirely cover the topic of error bounds and we thus refer the interested reader to Boucheron et al. (2005), Koltchinskii (2006), Bartlett et al. (2004), Bartlett and Mendelson (2006), Massart (2006), Massart (2000) and references therein for more information.

2.1 Notation

Random Variables and Distributions. We consider an input space $\mathcal{X}$, an output space $\mathcal{Y}$ and a probability distribution P on the product space $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$. Let $Z := (X,Y)$ ($Z \in \mathcal{Z}$) denote a pair of random variables distributed according to P and for a given integer n, let $Z_1,\dots,Z_n$ and $Z'_1,\dots,Z'_n$ be two independent samples of n independent copies of Z. We denote by P_n, P'_n and P_{2n} the empirical measures associated respectively to the first, the second and the union of both samples. $\mathbb{E}^n$, $\mathbb{E}'^n$ and $\mathbb{E}^{2n}$ denote the expectation with respect to the first, second and union of both training samples, while $\mathbb{P}^n, \mathbb{P}'^n$ and $\mathbb{P}^{2n}$ denote the distribution of these samples (i.e., $\mathbb{P}^n$ is the n-fold product distribution whose marginals are P).

Regret Functions. To each function $g : \mathcal{X} \to \mathcal{Y}$ and each function $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, we associate the function $f : \mathcal{Z} \to \mathbb{R}$ defined as

$$f(z) = L(g(x), y).$$

Such functions g, L and f will be respectively called prediction function, loss function and regret function. In classification, the loss function is $L = \mathbb{1}_{g(x) \neq y}$ where $\mathbb{1}$ denotes the indicator function. To each pair of prediction functions g_1 and g_2, we define the relative regret function $f : \mathcal{Z} \to \mathbb{R}$ defined as

$$f(z) = L(g_1(x), y) - L(g_2(x), y).$$

To each measurable real-valued function f defined on $\mathcal{Z}$, we denote their expectation under P by Pf and their empirical expectation by $P_n f$ $\big($i.e., $P_n f = n^{-1} \sum_{i=1}^n f(Z_i)\big)$. For (relative) regret functions, Pf is often called the (relative) risk.

Geometry of the Regret Class. Let $\mathcal{F}$ denote a set of measurable real-valued functions defined on $\mathcal{Z}$. Typical examples of such a class are based on regret function or relative regret functions and derived from a set of prediction functions. By slight extension, any set of measurable real-valued functions defined on $\mathcal{Z}$ will be called a regret class. On the set $\mathcal{F}$, we consider the pseudo-distance

$$d(f_1, f_2) = \sqrt{P(f_1 - f_2)^2},$$

and define similarly d_n, d_n' and d_{2n}. We define the covering number $N(\mathcal{F}, \varepsilon, d)$ as the minimum number of balls of radius ε needed to cover $\mathcal{F}$ in the pseudo-distance d.

An important task for obtaining error bounds is to define sieves, that is, subsets of $\mathcal{F}$ that approximate $\mathcal{F}$ to a certain distance. The simplest way is to use minimum covers at various scales, but it is also possible to use sequences of nested partitions as defined below.

Definition 1 *A sequence* $(\mathcal{A}_j)_{j \in \mathbb{N}}$ *is a* sequence of nested partitions *of* $\mathcal{F}$ *if*

- $\mathcal{A}_j$ *is a partition of* $\mathcal{F}$ *either countable or equal to the set of all singletons of* $\mathcal{F}$
- *The* $\mathcal{A}_j$ *are nested: each element of* $\mathcal{A}_{j+1}$ *is contained in an element of* $\mathcal{A}_j$*, and* $\mathcal{A}_0 = \{\mathcal{F}\}$

For a partition $\mathcal{A}$, we denote by $A(f)$ the unique element of $\mathcal{A}$ containing f.

Given a sequence of nested partitions $(\mathcal{A}_j)$, we can build a collection $(S_j)_{j \in \mathbb{N}}$ of approximating subsets of $\mathcal{F}$ in the following way: for each $j \in \mathbb{N}$, for each element A of $\mathcal{A}_j$, choose a unique element of $\mathcal{F}$ contained in A and define S_j as the set of all chosen elements. We have $|S_0| = 1$ and denoting by $p_j(f)$ the unique element of S_j contained in $A_j(f)$ we have

$$\begin{cases} p_j(f) = f \text{ for any } f \in S_j, \\ p_{j-1} \circ p_j = p_{j-1}. \end{cases}$$

Measures on the Regret Class. We denote by ρ and π two probability measures on the space $\mathcal{F}$, so that $\rho P f$ will actually mean the expectation of Pf when f is sampled according to the probability measure ρ. We will denote by $\mathcal{M}_+^1(\mathcal{F})$ the set of all probability measures on $\mathcal{F}$. For two such measures, $K(\rho, \pi)$ will denote their Kullback-Leibler divergence defined as

$$K(\rho, \pi) = \rho \log \frac{d\rho}{d\pi} := \int_{\mathcal{F}} \log\left(\frac{d\rho}{d\pi}(f)\right) \rho(df),$$

when ρ is absolutely continuous with respect to π and $K(\rho, \pi) = +\infty$ otherwise.

Other Notation. The notation x_+ and x_- refers to the positive and negative parts respectively, that is, $x_+ := \max(x, 0)$ and $x_- := \max(-x, 0)$. β will denote some positive real number while C will be some positive constant (whose value may differ from line to line).

2.2 The Setting for Error Bounds

Generalization error bounds give upper bounds on the true (i.e., under P) error of the function returned by a learning algorithm. These upper bound typically involve the empirical error of this function. A learning algorithm (or learning rule), is a map from sequences of pairs $(X_1,Y_1),\dots,(X_n,Y_n)$ to functions $g : \mathcal{X} \to \mathbb{R}$. We will denote by g_n the function returned by the algorithm under investigation.

2.2.1 ABSOLUTE AND RELATIVE RISK BOUNDS

The goal is thus to formulate a statement of the following form: for each (small enough) $\beta > 0$, with probability at least $1-\beta$ with respect to the random draw of the sample,

$$\mathbb{E}\left[L(g_n(X),Y)|(X_1,Y_1),\dots,(X_n,Y_n)\right] \le \frac{1}{n}\sum_{i=1}^{n} L(g_n(X_i),Y_i) + B(n,\beta)\,, \tag{1}$$

where the expectation in the left-hand side is taken with respect to the distribution of (X,Y), meaning: conditionally to the training sample distribution.

Another type of result that can be obtained is a relative risk bound where one compares the risk of the algorithm to the risk of a fixed prediction function $\tilde{g}$. The type of inequality that could be obtained looks like this

$$\begin{aligned}\mathbb{E}\left[L(g_n(X),Y)|(X_1,Y_1),\dots,(X_n,Y_n)\right] - \mathbb{E}\left[L(\tilde{g}(X),Y)\right]\\ \le \tfrac{1}{n}\textstyle\sum_{i=1}^{n}\left[L(g_n(X_i),Y_i) - L(\tilde{g}(X_i),Y_i)\right] + B(n,\beta)\,.\end{aligned} \tag{2}$$

With the notation introduced above, denoting by f_n the (relative) regret function associated to the predicition g_n we can rewrite both previous inequalities as follows:

$$Pf_n - P_n f_n \le B(n,\beta)\,. \tag{3}$$

This shows that there is no essential difference between the techniques used to get bounds of the form (1) and (2).

2.2.2 AVERAGED BOUNDS

Most algorithms simply pick a candidate function g_n from a fixed set $\mathcal{G}$. However, Bayesian algorithms usually aggregate several such functions by taking their weighted average. The corresponding regret function is not the average of the regrets of the individual functions which makes the analysis difficult. In order to avoid this caveat, it is common to first study randomized estimators and then relate the randomized to the aggregate ones. Randomized estimators are built by replacing the weighted average by a randomized choice (where the probability of choosing a specific function is proportional to its weight). The advantage of such a procedure is that it is relatively simple to analyze. Indeed, if one uses weights given by a probability distribution ρ on $\mathcal{G}$, the error of the randomized estimator is simply the average of the errors of the combined estimators ρPf.

Hence, bounds for randomized estimators will have the form

$$\rho_n Pf \leq \rho_n P_n f + B(n,\beta), \tag{4}$$

where ρ_n is the specific distribution chosen by the algorithm based on the data.

2.2.3 ALGORITHM AND DATA DEPENDENT BOUNDS

In the form stated in (3), the quantity B only depends on the confidence level β and the sample size n. The only way in which B depends on the algorithm is usually by incorporating terms that depend on the class of prediction functions used by the algorithm. For example if the algorithm picks functions in a class $\mathcal{G}$ whose associated regret class is $\mathcal{F}$, statement (3) can be deduced from

$$\sup_{f\in\mathcal{F}} \left\{Pf - P_n f\right\} \leq B(n,\beta).$$

This is a *supremum bound* and is the most common type found in the literature. Unfortunately, this usually yields quantitatively loose bounds which do not tell much about the algorithm's behavior.

Ideally, B should depend on the sample (data-dependent bound) and on the specific function f_n or distribution ρ_n chosen by the algorithm (algorithm-dependent bound).

Hence we can aim at obtaining bounds of the form (algorithm-dependent)

$$\forall f \in \mathcal{F}, Pf \leq P_n f + B(n,\beta,f),$$

or even (algorithm and data-dependent)

$$\forall f \in \mathcal{F}, Pf \leq P_n f + B(n,\beta,f,Z_1,\ldots,Z_n).$$

2.3 Previous Error Bounds

We are now in a position to state and compare some of the previously known error bounds. From now on, the functions in the regret class $\mathcal{F}$ are assumed to be bounded. Without loss of generality, we may assume that they take their values in $[0;1]$. We do not always state the bounds in their original form in order to allow an easier comparison and we thus include the proofs and the explicit values of the constant C in Section B.3.

2.3.1 SUPREMUM BOUNDS

We start with the most common bounds involving the supremum over a class of the difference between true and empirical error.

Single function. The starting point is to consider a class containing only one function f. By Hoeffding's inequality one easily gets that with probability at least $1-\beta$,

$$Pf - P_n f \leq C\sqrt{\frac{\log(\beta^{-1})}{n}}. \tag{5}$$

Unfortunately, when $\mathcal{F}$ has more than one element, this statement can only be made separately for each function. [Proof in Section B.3.1]

Finite union bound. It is easy to convert the above statement into one which is valid simultaneously for a finite set of functions $\mathcal{F}$. The simplest form of the union bound gives that with probability at least $1-\beta$,

$$\sup_{f\in\mathcal{F}}\left\{Pf-P_nf\right\}\le C\sqrt{\frac{\log|\mathcal{F}|+\log(\beta^{-1})}{n}}. \tag{6}$$

The term $\log|\mathcal{F}|$ represents the extra price to pay to obtain a uniform statement. It can be considered as a measure of the complexity of the class $\mathcal{F}$. [Proof in Section B.3.2]

Symmetrization. When $\mathcal{F}$ is infinite this cannot work directly. The trick is to introduce a second sample $Z_1',\dots,Z_n'$ (see the notation section for definitions) and to consider the set of vectors formed by the values of each function in $\mathcal{F}$ on the double sample. When the functions have values in $\{0;1\}$, this is a finite set and the above union bound applies. This idea was first used in Vapnik and Chervonenkis (1971) to obtain that with probability at least $1-\beta$,

$$\sup_{f\in\mathcal{F}}\left\{Pf-P_nf\right\}\le C\sqrt{\frac{\log\mathbb{E}^{2n}N(\mathcal{F},1/2n,d_{2n})+\log(2\beta^{-1})}{n}}. \tag{7}$$

The capacity is here better estimated than in (6) since the term $N(\mathcal{F},1/2n,d_{2n})$ does not count twice functions classifying in the same way the training and virtual samples. However the quantity $\mathbb{E}^{2n}N(\mathcal{F},1/2n,d_{2n})$ cannot be computed in general since P is unknown, but upper bounds can be obtained in terms of combinatorial parameters such as the VC dimension.

We now see that the complexity of $\mathcal{F}$ has to be measured in a way that involves the metric induced by the distribution. What matters is thus not how many functions there are in $\mathcal{F}$ but how they span the space of possible vectors $(f(Z_1),\dots,f(Z_n),f(Z_1'),\dots,f(Z_n'))$, which can be measured by the minimum number of balls required to get an approximation at scale $1/2n$. [Proof in Section B.3.3]

Chaining. One limitation of the above result is that it applies only to $\{0;1\}$-valued functions and it measures the size only at the smallest scale which is known to be suboptimal in general. The union bound can be refined by considering finite covers of the set of function at different scales. This is called the *chaining* technique, pioneered by Dudley (1984) since one constructs a chain of functions that approximate a given function more and more closely. The results involve the Koltchinskii-Pollard entropy integral as, for example in Devroye and Lugosi (2001). One has with probability at least $1-\beta$,

$$\sup_{f\in\mathcal{F}}\left\{Pf-P_nf\right\}\le C\left(\frac{1}{\sqrt{n}}\mathbb{E}^n\int_0^\infty\sqrt{\log N(\mathcal{F},\varepsilon,d_n)}d\varepsilon+\sqrt{\frac{\log(\beta^{-1})}{n}}\right). \tag{8}$$

[Proof in Section B.3.8]. Chained bounds are particularly useful when one has a tight control of the entropy numbers of the set $\mathcal{F}$ (see, for example, van der Vaart 1998 and van de Geer 2000).

Generic chaining. It has been noticed by Fernique and Talagrand that it is possible to capture the complexity in a better way than using minimal covers by considering majorizing measures or generic chaining. This is essentially optimal for Gaussian processes and optimal up to logarithmic factors for empirical processes. Let $r>1$ and $(\mathcal{A}_j)_{j\ge1}$ be a sequence of nested partitions of $\mathcal{F}$ such that all the elements of $\mathcal{A}_j$ have diameter at most r^{-j} w.r.t. the distance d_n. Let also $(\pi^{(j)})$ be a sequence of probability distributions defined on $\mathcal{F}$. We can prove that with probability at least $1-\beta$,

$$\sup_{f\in\mathcal{F}}\{Pf-P_nf\}\leq C\left(\frac{1}{\sqrt{n}}\mathbb{E}^n\sup_{f\in\mathcal{F}}\sum_{j=1}^{\infty}r^{-j}\sqrt{\log\{1/\pi^{(j)}[A_j(f)]\}}+\sqrt{\frac{\log(\beta^{-1})}{n}}\right). \qquad (9)$$

[Proof in Section B.3.9]. The interpretation of the complexity term is a bit harder now.[1] If one takes partitions induced by minimal covers of $\mathcal{F}$ at radii r^{-j}, and uniform measures concentrated at the centers of the balls, one has $1/\pi^{(j)}[A_j(f)]=N(\mathcal{F},r^{-j},d_n)$ so that (9) leads to a sum of terms of the form $r^{-j}\sqrt{\log N(\mathcal{F},r^{-j},d_n)}$, which allows to recover (8). This means that the complexity term of (9) is at least as sharp as the one of (8), but has more flexibility since the partitions can be built in a better way than via uniform covers.

As in Section 2.1, from the nested partitions $(\mathcal{A}_j)$, we can build a collection $(S_j)_{j\in\mathbb{N}}$ of approximating subsets of $\mathcal{F}$ in the following way: for each $j\in\mathbb{N}$, for each element A_j of $\mathcal{A}_j$, choose a unique element of $\mathcal{F}$ contained in A_j and define S_j as the set of all chosen elements. Then consider $p_j(f)$ the unique element of S_j contained in $A_j(f)$. The $p_j(f)$ define successive approximations of f. Without loss of generality, consider that the null function belongs to $\mathcal{A}_0$ and that S_0 contains this function. The sum in (9) (as the integral in (8)) comes from the core of the chaining idea which is to decompose the function f (which is a difference between two functions in the case of relative regret classes, or a difference between itself and the null function in the case of a regret class) into $\sum_{j>0}p_j(f)-p_{j-1}(f)$ and to separately control the deviation of each term. These terms form a *chain* of finer and finer approximations to f that give the name to the method (chaining). This is essentially the idea that we use in the proof of our main results.

Let us try to give some insights about how to construct the partitions $(\mathcal{A}_j)$ (for more details, see Talagrand, 2005). First of all, one should understand that the measures $\pi^{(j)}$ play no essential role here (as was noticed by Talagrand, 2001, , they can be taken as finitely supported uniform measures on appropriate subsets) The only reason why we state the result in the above form is to emphasize the connection with our main result (to be presented in Section 3).

It turns out that there are many ways to state the majorizing measure/generic chaining bound. Each way involves a different geometric construction on the regret class $\mathcal{F}$ and a different notion of size, but most can be shown to be equivalent up to constant factors. The general form of such bounds is

$$\sup_{f\in\mathcal{F}}\sum_{i\geq i_0}F(f,i)G(f,i)$$

where $F(f,i)$ is a measure of the scale of the geometric object of order i containing f, while $G(f,i)$ is a measure of the size of this object. For example, if one uses balls, $F(f,i)$ is the radius of the ball $\left(\text{e.g., } F(f,i)=2^{-i}\right)$ and $G(f,i)$ is the "mass" (or rather a function of the mass) of the ball centered at f and of radius 2^{-i} $\left(\text{e.g., } G(f,i)=\sqrt{\log 1/\pi^{(i)}[B(f,2^{-i})]}\right)$.

1. It is important to mention that there are various possible ways to measure the complexity of the space $\mathcal{F}$ that all lead to an essentially optimal result. For example, one could replace the set $A_j(f)$ by a ball centered at f and with radius r^{-j}. This would give the standard majorizing measure bound (where all the $\pi^{(j)}$ are the same). Also, one can use partitions whose elements are allowed to have different diameters and replace the r^{-j} term by the diameter of $A_j(f)$. Using such an approach actually allows to get rid of the measures $\pi^{(j)}$, the sum in the complexity term becoming $\sum d(A_j(f))\sqrt{\log|\mathcal{A}_j|}$. We did not choose this formulation as we wish to obtain a result that explicitly uses the probability π so that it can be more directly related to the PAC-Bayesian bounds. The interested reader is referred to Talagrand (2005) for more information about the variants of the generic chaining bounds.

Coming back to nested partitions (which really are the key ingredient here), let us say a few words about their construction. Talagrand proposed a partitioning scheme that runs as follows. The general idea is that one starts with a size function (that measures how big a subset of $\mathcal{F}$ is). This function has to satisfy certain compatibility conditions in order to ensure that the resulting construction will be optimal (up to constants). Then one starts with $\mathcal{A}_0 = \{\mathcal{F}\}$ and, at each stage, the partition is refined by splitting each of its element in a greedy way: one selects separated enough points in the element to be split and builds balls arounds these points. Starting with the "biggest" ball (as measured by the size function), one removes the balls until nothing is left. The sub-partition elements are thus the subsets that are removed with each ball.

Rademacher averages. As we said before, the generic chaining bound is optimal up to constant factors. More precisely, it is a tight upper bound for the quantity

$$\mathbb{E}^n \left[\sup_{f \in \mathcal{F}} \left\{ Pf - P_n f \right\} \right].$$

It is also a lower bound for this quantity but up to a $\log n$ factor. Hence the generic chaining bound is as good (up to this log) as another well-known quantity in the learning theory community, the Rademacher average of $\mathcal{F}$:

$$\mathbb{E}^n \left[\mathbb{E}_\sigma \sup_{f \in \mathcal{F}} \frac{1}{n} \sum \sigma_i f(Z_i) \right],$$

where the σ_i are independent random signs ($+1, -1$ with probability $1/2$). Using this quantity as a measure of complexity, it is possible to obtain the following statement. One has with probability at least $1-\beta$,

$$\sup_{f \in \mathcal{F}} Pf - P_n f \leq C \left(\frac{1}{n} \mathbb{E}^n \mathbb{E}_\sigma \sup_{f \in \mathcal{F}} \sum_{i=1}^n \sigma_i f(Z_i) + \sqrt{\frac{\log(\beta^{-1})}{n}} \right). \tag{10}$$

[Proof in Section B.3.7]

2.3.2 VARIANCE (LOCALIZED) BOUNDS

It has been noticed that the supremum bounds do not give optimal rates of convergence for algorithms based on the minimization of the empirical risk. The main reason is that the size of the deviations between Pf and $P_n f$ depends on the variance $\operatorname{Var} f$. A lot of work has been carried out recently in order to take this into account in the bounds presented above, for example by restricting the complexity term to functions with small variance.

Variance for Single Functions. Since the deviations between Pf and $P_n f$ for a given function f actually depend on its variance, one can refine (5) into

$$Pf - P_n f \leq C \left(\sqrt{\frac{\operatorname{Var} f \log(\beta^{-1})}{n}} + \frac{\log(\beta^{-1})}{n} \right). \tag{11}$$

[Proof in Section B.3.5]

Variance with Symmetrization. The above inequality can be combined with the symmetrization trick. This was done by Vapnik and Chervonenkis (1974) (for functions in $\{0;1\}$). Their result is that with probability at least $1-\beta$,

$$\forall f \in \mathcal{F}, Pf - P_n f \leq C\sqrt{Pf}\sqrt{\frac{\log \mathbb{E}^{2n} N(\mathcal{F}, 1/2n, d_{2n}) + \log(4\beta^{-1})}{n}}, \tag{12}$$

which also gives with probability at least $1-\beta$,

$$\forall f \in \mathcal{F}, Pf - P_n f \leq C\Big(\sqrt{P_n f}\sqrt{\tfrac{\log \mathbb{E}^{2n} N(\mathcal{F},1/2n,d_{2n})+\log(4\beta^{-1})}{n}} + \tfrac{\log \mathbb{E}^{2n} N(\mathcal{F},1/2n,d_{2n})+\log(4\beta^{-1})}{n}\Big).$$

One can thus consider that the capacity term is weighted by the risk (or equivalently here by the empirical risk).

Localized Rademacher Averages. It is also possible to combine (11) with (10) as was done for example by Bartlett et al. (2005). This gives a complexity term which involves Rademacher averages that are computed on subsets of functions with small variance.

How to Use Variance Bounds. In order to better explain how variance bounds can be used efficiently, we should notice that the quantitative gain occurs only when $\operatorname{Var} f$ is small for the function chosen by the algorithm.

For a loss function L taking its values $[0;1]$, the variance of a regret function f can be bounded successively by Pf^2 and Pf. Consequently, in low noise setting (i.e., when there exists a prediction function such that its associated risk Pf is small), one can expect that the deviation of the risk of an algorithm based on the minimization of the empirical error is much smaller than in noisy situations.

For relative regret function the situation is different. We are still interested in low (relative) risk but in general we will not have any particular relation between the variance $\operatorname{Var} f$ and the relative risk Pf. Consequently variance localized bounds will not yield any significant improvement. However in some situations as in classification under Mammen and Tsybakov noise condition (Tsybakov, 2004) or in least square regression (see, for example, Audibert, 2004a), one has the inequality $\operatorname{Var} f \leq (Pf)^\alpha$ for some positive α. This inequality is well exploited by variance localized bounds.

2.3.3 ALGORITHM DEPENDENT COMPLEXITY

Another direction in which the bounds can be improved is by making the bound depend more directly of the function chosen by the algorithm. The variance bounds already have this property but the bound only depends on $\operatorname{Var} f$ but not on where the function is in the space $\mathcal{F}$. We now present results where the complexity term depends directly on the selected function.

Weighted union bound and algorithm dependence. The finite union bound can be directly extended to the countable case by introducing a probability distribution π over $\mathcal{F}$ which weights each function (McAllester, 1998) and gives, with probability at least $1-\beta$,

$$\forall f \in \mathcal{F}, Pf - P_n f \leq C\sqrt{\frac{\log 1/\pi(f) + \log(\beta^{-1})}{n}}. \tag{13}$$

The complexity term now depends on f and of course, taking a uniform distribution on a finite set we recover (6).

Surprisingly, the capacity term could be arbitrarily small if π were chosen appropriately. However, π has to be chosen before seeing the data, so there is no way to ensure that the bound will be always small.

It is important to understand that the choice of π is completely arbitrary and need not reflect any prior belief in what is the true target function. The distribution π is just a "technical" prior which is used to formulate the bound.

Inequality (13) can be read as follows: given a fixed π, if one samples data repeatedly, with high probability, the error of any function f in the class will be upper bounded by a function of $\pi(f)$. This function of $\pi(f)$ is basically a bound on the deviation one would expect for each individual function, plus an extra term coming from the fact that the function f to be considered is unknown prior to observing the sample so that we need to have a statement which holds simultaneously for all functions in the class. The interesting point is that in making this statement uniform, there is a freedom in the choice of how we distribute the 'extra cost'. The distribution of the uniformization cost is thus represented by π. Any probability distribution will do, and if one is lucky, the functions of interest (the ones returned by the classification algorithm when given typical samples from the problem) will have large prior and the bound for them will be relatively small.

Thus, if one wants to obtain a bound which has small values, one has to "guess" how likely each function $f \in \mathcal{F}$ is to be chosen by the algorithm.

Averaging. We now come to the case of randomized predictions which give rise (as explained above) to averaged bounds of the form (4). Consider a probability distribution ρ_n defined on a countable $\mathcal{F}$, take the expectation of (13) with respect to ρ_n and use Jensen's inequality. This gives with probability at least $1-\beta$,

$$\rho_n(Pf - P_nf) \leq C\sqrt{\frac{K(\rho_n,\pi)+H(\rho_n)+\log(\beta^{-1})}{n}},$$

where ρ_n is still the specific distribution chosen by the algorithm based on the data and $H(\rho_n)$ is its Shannon entropy. The l.h.s. is the difference between true and empirical error of a randomized classifier which uses ρ_n as weights for choosing the decision function (independently of the data). The following PAC-Bayes bound (McAllester, 1999) refines the above bound since it has the form (for possibly uncountable $\mathcal{F}$)

$$\rho_n(Pf - P_nf) \leq C\sqrt{\frac{K(\rho_n,\pi)+\log n+\log(\beta^{-1})}{n}}. \tag{14}$$

This in particular shows that the entropy term is unnecessary. To some extent, one can consider that the PAC-Bayes bound is a refined union bound where the gain happens when ρ_n is not concentrated on a single function (or more precisely ρ_n has entropy larger than $\log n$).

The complexity now depends upon the arbitrary choice of π and one may notice that it is modulated by the "spread" of ρ_n. Indeed, if ρ_n is concentrated, this term can be big, but if ρ_n is similar to π it becomes very small. As a special case, if the randomizing distribution is concentrated at a single function (corresponding to classical algorithms that simply pick a function), the bound has the form (13) which is not suited for large (e.g., uncountable) sets of functions.

2.3.4 DATA-DEPENDENT BOUNDS

We now consider ways to obtain bounds where the complexity term itself depend on the sample (hence can be computed from the data only).

Transductive priors. It is actually possible to combine the symmetrization and weighting ideas (Catoni, 2003). For example, if one defines a function $\Pi : \mathcal{Z}^{2n} \to \mathcal{M}_+^1(\mathcal{F})$ that is *almost exchangeable* in the sense that if we exchange the value of z_i and z_{i+n} we do not change the value of the function, then one gets, with probability at least $1-\beta$ (over the random choice of a double sample),

$$\forall f \in \mathcal{F}, P'_n f - P_n f \leq C\sqrt{\frac{\log 1/\Pi(Z_1,\ldots,Z_n,Z'_1,\ldots,Z'_n)(f) + \log(\beta^{-1})}{n}}.$$

Note that we no longer require $\mathcal{F}$ to be countable but $\Pi(.)$ should have countable support for each value of the double sample. This type of result is interesting in the transduction framework, where the instances (or inputs) are known in advance and where the randomness is in the way the data is split into a training set (with known labels) and a testing set (with unknown labels). Converting this into an induction statement (comparing the empirical with the expected errors) gives with probability $1-\beta$,

$$\forall f \in \mathcal{F}, Pf - P_n f \leq C\mathbb{E}'^n\sqrt{\frac{\log 1/\Pi(Z_1,\ldots,Z_n,Z'_1,\ldots,Z'_n)(f) + \log(\beta^{-1})}{n}}.$$

This is useful provided one can upper bound the first logarithmic factor in the r.h.s. either with a data-independent quantity or with an observable function of the first sample.

Concentration. Using concentration inequalities as in Boucheron et al. (2000) for example, one can get rid of the expectation appearing in the r.h.s. of (7), (8), (10) or (9) and thus obtain a bound that can be computed from the data.

In particular, a data-dependent and localized version of (10) is given in Bartlett et al. (2005). However it has not been combined with the PAC-Bayes improvement for randomized predictions.

2.3.5 SUMMARY

Our goal in this work is to see how to attempt to combine the different approaches that have been used before, in the hope to obtain a bound that has the combined properties

1. structural (metric structure effect)

2. statistical (variance effect)

3. PAC-Bayesian (averaging effect).

The connection between 1 and 2 already exists, so does the connection between 2 and 3. Our main result aims at dealing simultaneously with 1, 2 and 3.

The main difficulty in connecting 1 and 2 with 3 is that if one has a non-countable infinite set of functions, even using symmetrization, if the prior π is non-atomic, then $\pi(\{f\}) = 0$ for all f. Hence the complexity term $K(\rho_n,\pi)$ blows up when ρ_n is concentrated on a single function. The result we present below is, to our knowledge, the first one which actually bridges this gap since the complexity term does not blow up when ρ_n is concentrated at one function.

3. Main results

We now state and comment our main result. We recall that $\mathcal{F}$ denotes a set of functions defined on a measurable space $\mathcal{Z}$ and taking their values in $[0;1]$. In the theorem we present, one has to

- choose a sequence of nested partitions $(\mathcal{A}_j)_{j\in\mathbb{N}}$ of the set $\mathcal{F}$,
- build an associated sequence of approximating sets $(S_j)_{j\in\mathbb{N}}$ (see Section 2.1),
- for each f and $j \in \mathbb{N}$, define its approximating functions $p_j(f)$ as the unique element of S_j contained in $A_j(f)$, where $A_j(f)$ is the set of the partition $\mathcal{A}_j$ containing f
- for each $j \in \mathbb{N}$, choose the distribution $\pi^{(j)}$ on $\mathcal{F}$.

The quantities $\pi^{(j)}$, S_j, and p_j are allowed to depend on the sample $Z_1,\dots,Z_n$ provided that it also depends on a double sample $Z'_1,\dots,Z'_n$ in an almost exchangeable way (i.e., exchanging Z_i and Z'_i does not affect their value). Denote δ_f the Dirac measure on f. For a probability distribution ρ on $\mathcal{F}$, define its j-th projection as

$$[\rho]_j = \sum_{f\in S_j} \rho[A_j(f)]\delta_f,$$

when S_j is countable and $[\rho]_j = \rho$ otherwise. When S_j is countable, $[\rho]_j$ is a probability distribution on $\mathcal{F}$ supported by S_j and can be viewed as the projection of ρ on S_j. Let ρ_n be a randomized estimator, that is, a data-dependent distribution on the set $\mathcal{F}$. To shorten the notation, the average (w.r.t. the probabiblity distribution ρ_n) distance between two successive approximations is denoted by

$$\rho_n d_j^2 := \rho_n d_{2n}^2[p_j(f), p_{j-1}(f)],$$

where we recall that d_{2n} is the empirical pseudo-distance $d_{2n}(f_1,f_2) = \sqrt{P_{2n}(f_1-f_2)^2}$. Finally, let

$$\Delta_{n,j}(f) := P'_n[f-p_j(f)] - P_n[f-p_j(f)],$$

$$\chi(x) := \sqrt{x}\log\log\left(4e^2/x\right),$$

and for $\beta > 0$ introduce

$$K_j := K([\rho_n]_j, [\pi^{(j)}]_j) + \log[j(j+1)\beta^{-1}] \tag{15}$$

in which the leading term is the Kullback-Leibler divergence between the j-th projections of the randomized distribution ρ_n and the (j-th prior) distribution $\pi^{(j)}$.

The previous definitions are common to both transduction and induction setting. However the double sample is a totally virtual one in the induction setting.

We consider first the so-called transduction setting, or rather, a version of it. In this setting, we recall that the two independent samples $Z_1,\dots,Z_n$ and $Z'_1,\dots,Z'_n$ are drawn i.i.d. according to the unknown probability P.

The learning algorithm is allowed to use the instances of both samples (training and testing samples) but has access to the labels of the training instances only. Its goal is to correctly predict the instances of the testing samples. In this context, the quantity of interest is the difference between the average misclassification error obtained on the testing sample and the one on the training sample.

Theorem 2 (Transduction) *Let* $0 < \beta \leq 2e^{-1}$. *If the following condition holds*

$$\lim_{j\to+\infty} \sup_{f\in\mathcal{F}} \Delta_{n,j}(f) = 0, \qquad \mathbb{P}^{2n}\text{-}a.s. \tag{16}$$

then, with $\mathbb{P}^{2n}$*-probability at least* $1-\beta$*, for any distribution* ρ_n*, we have*

$$\rho_n P'_n f - P'_n f_0 \le \rho_n P_n f - P_n f_0 + \frac{2\sqrt{2}e^{\frac{1}{4}}}{\sqrt{n}} \sum_{j=1}^{+\infty} \sqrt{K_j \rho_n d_j^2} + \frac{2\sqrt{2}e^{\frac{1}{4}}}{\sqrt{n}} \sum_{j=1}^{+\infty} \chi\Big(\frac{\rho_n d_j^2}{K_j}\Big).$$

In the induction setting, the learning algorithm is only allowed to use the first sample $Z_1,\dots,Z_n$. Nevertheless the proof technique uses an independent and i.i.d. virtual sample $Z'_1,\dots,Z'_n$ that is at the origin of the following expectations $\mathbb{E}'^n$.

Theorem 3 (Induction) *With the above notation, if the following condition holds*

$$\limsup_{j\to+\infty} \sup_{f\in\mathcal{F}} \mathbb{E}'^n \Delta_{n,j}(f) \le 0, \qquad \mathbb{P}^n\text{-}a.s. \tag{17}$$

then for any $0<\beta\le 0.73$*, with* $\mathbb{P}^n$*-probability at least* $1-\beta$*, we have*

$$\rho_n Pf - Pf_0 \le \rho_n P_n f - P_n f_0 + \tfrac{3.7}{\sqrt{n}} \textstyle\sum_{j=1}^{+\infty} \sqrt{\mathbb{E}'^n K_j \mathbb{E}'^n \rho_n d_j^2} + \tfrac{3.7}{\sqrt{n}} \textstyle\sum_{j=1}^{+\infty} \chi\Big(\frac{\mathbb{E}'^n \rho_n d_j^2}{\mathbb{E}'^n K_j}\Big).$$

Remark 1 *The second sum in the bound is in general negligible w.r.t. the first one*[2] *and has at worse the same order.*

Remark 2 *Let* $\mathcal{G}$ *be a model (i.e., a set of prediction functions). Let* $\tilde{g}$ *be a reference function (not necessarily in* $\mathcal{G}$ *and possibly depending on the data in an exchangeable way). Consider the class of regret functions* $\mathcal{F} = \{z \mapsto L[g(x),y] : g \in \mathcal{G} \cup \{\tilde{g}\}\}$*. Define* $f_0 = L[\tilde{g}(x),y]$*. The induction (resp. transduction) theorem compares the risk (resp. the risk on the second sample) of any (randomized) estimator with the risk (resp. the risk on the second sample) of the reference function* $\tilde{g}$*.*

Remark 3 *Assumption* (16) *is not very restrictive. For instance, it is satisfied when one of the following condition holds:*

- *there exists* $J \in \mathbb{N}^*$ *such that* $S_J = \mathcal{F}$*,*
- *almost surely* $\lim_{j\to+\infty} \sup_{f\in\mathcal{F}} \|f - p_j(f)\|_\infty = 0$ *(it is in particular the case when the bracketing entropy of the set* $\mathcal{F}$ *is finite for any radius and when the* S_j*'s and* p_j*'s are appropriately built on the bracketing nets of radius going to* 0 *when* $j\to+\infty$*),*
- *almost surely* $\lim_{j\to+\infty} \sup_{f\in\mathcal{F}} d_{2n}(f, p_j(f)) = 0$.[3]

Finally, by Lebesgue's dominated convergence theorem and standard probabilistic arguments, one may prove that (16) *implies* (17).

Remark 4 *Note that* $\mathbb{E}'^n \rho_n d_j^2 = \frac{1}{2}\{\rho_n d_n^2[p_j(f), p_{j-1}(f)] + \rho_n d^2[p_j(f), p_{j-1}(f)]\}$*. Besides, when the quantities* $\pi^{(j)}$*,* S_j*,* f_0 *and* p_j *do not depend on the data, we have* $\mathbb{E}'^n K_j = K_j$*, so that the conditional expectation* $\mathbb{E}'^n$ *disappears in Theorem 3.*

Remark 5 *A slightly different version of Theorem 3 can be obtained by using Theorem 2 and Lemma 19.*

2. Since K_j is greater than or equal to 1 and the function χ is an upper bounded function which behaves as the square root near 0.
3. Here we give a typical construction for which this assumption is satisfied when the functions in $\mathcal{F}$ take their values in a common finite set (e.g., classification setting). Take the sequence S_j as embedded nets w.r.t. the pseudo-distance d_{2n} of radius tending to 0 when j goes to infinity. Then there exists J satisfying $\big|\big(f(X_1),\dots,f(X_n),f(X'_1),\dots,f(X'_n)\big) : f \in S_J\big| = \big|\big(f(X_1),\dots,f(X_n),f(X'_1),\dots,f(X'_n)\big) : f \in \mathcal{F}\big|$ and $S_j = S_J$ for any $j > J$. Consider projections p_j consistent with d_{2n} to the extent that, if $d_{2n}(f, S_j) = 0$, then $d_{2n}(f, p_j(f)) = 0$. Then the assumption holds.

4. Discussion

We now present in which sense the result presented above combines several previous improvements in a single bound. The discussion will also clarify how to choose the priors $\pi^{(j)}$, and the sets S_j by giving more explicit bounds for some particular choices.

4.1 Supremum Bounds

We first show that we can derive from Theorem 3 a result similar to the generic chaining bound (9).

Corollary 4 *Under Assumption* (17)*, we have with* $\mathbb{P}^n$*-probability at least* $1-\beta$*,*

$$\forall f \in \mathcal{F}, Pf - P_n f - Pf_0 + P_n f_0 \\ \leq C\left(\frac{1}{\sqrt{n}}\sum_{j\geq j(f)}^{\infty} 2^{-j}\sqrt{\mathbb{E}'^n \log\{1/\pi^{(j)}[A_j(f)]\}} + \sqrt{\frac{\log 2j(f)+\log(\beta^{-1})}{2^{j(f)}n}}\right),$$

where $j(f) := \min\{j \in \mathbb{N}^* : p_j(f) \neq f_0\}$.

Proof Choose for ρ_n a distribution concentrated at the single function f_n. Then we have $[\rho_n]_j = \delta_{p_j(f_n)}$ and K_j, defined in (15), reduces to $\log\{1/\pi^{(j)}[A_j(f_n)]\} + \log[j(j+1)\beta^{-1}]$. We can take the partitions $\mathcal{A}_j$ to have diameter 2^{-j} so that $\sup_{f\in\mathcal{F}} d_{2n}[f, p_j(f)] \leq 2^{-j}$ and thus $d_j \leq 2^{-j+1}$. ■

The "closer" the function f is from f_0, the bigger the integer $j(f)$ is. This result improves on (9) since it is algorithm dependent (the l.h.s. depends on f_n) and takes into account the variance. Since the series starts from $j(f)$, our bound is better when f is "close" to f_0.

Since we essentially have a result that is as powerful as generic chaining, it should be possible to recover Rademacher averages bounds. However, there is a difficulty coming from the fact that the result we have is not "symmetric" in the sense that it involves taking expectations over the second sample. Taking care of this remains a topic for further research.

4.2 Variance Bounds

We now show how the variance of the function of interest can be obtained explicitly in the upper bound. In particular, we have the following corollary.

Corollary 5 *Under Assumption* (17)*, if the functions in* $\mathcal{F}$ *have values in* $\{0;1\}$*, we have with* $\mathbb{P}^n$*-probability at least* $1-\beta$*,*

$$\forall f \in \mathcal{F}, Pf - P_n f - Pf_0 + P_n f_0 \\ \leq \frac{C}{\sqrt{n}}\sqrt{\mathbb{E}'^n P_{2n}(f-f_0)^2(\mathbb{E}'^n N(\mathcal{F}, 1/2n, d_{2n}) + \log(\beta^{-1}))}.$$

Proof As before, choose for ρ_n a distribution concentrated at a single function f_n, then $[\rho_n]_j = \delta_{p_j(f_n)}$ and thus $K_j = \log 1/\pi^{(j)}[A_j(f_n)]$. Now since the functions are binary valued, $\mathcal{F}$ is a finite metric space under the metric d_{2n}. We can thus take $\mathcal{A}_1$ as a minimal cover at radius $1/2n$, $\mathcal{A}_2 = \mathcal{F}$ and we will get $d_2(f) = 0$ and $d_1(f) = P_{2n}(f-f_0)^2$. Taking $\pi^{(j)}$ to be uniform on the centers of the cover, we will obtain $K_1 = \log N(\mathcal{F}, 1/2n, d_{2n}) + \log(\beta^{-1})$ which gives the result. ■

We thus essentially recover (12) and a fortiori standard VC bounds.

4.3 Averaging

Theorem 2 also includes the PAC-Bayesian improvement for averaging classifiers since if one considers the set $S_1 = \mathcal{F}$ one recovers a result similar to McAllester's (14). More precisely, we obtain:

Corollary 6 *Let $\mathcal{F}$ be a set of functions taking their values in $[-1;1]$. Let $0 < \beta \leq 0.73$ and $\mathcal{K} := K(\rho_n, \pi) + \log(2\beta^{-1})$. With $\mathbb{P}^{2n}$-probability at least $1-\beta$, we have*

$$\rho_n P'_n f - \rho_n P_n f \leq \tfrac{3.7}{\sqrt{n}} \sqrt{\mathcal{K} \rho_n P_{2n} f^2} + \tfrac{3.7}{\sqrt{n}} \chi\left(\tfrac{\rho_n P_{2n} f^2}{\mathcal{K}}\right).$$

As a consequence, for any $\beta > 0$, with $\mathbb{P}^n$-probability at least $1-\beta$, we have

$$\rho_n P f - \rho_n P_n f \leq C \sqrt{\tfrac{K(\rho_n, \pi) + \log(2\beta^{-1})}{n}}.$$

Note that this last inequality is slightly better than (14) since there is no extra logarithmic term. However the cost of removing the logarithmic factor is to have a slightly bigger constant C.

Proof Even if it means expanding $\mathcal{F}$, we may assume that the function f_0 identically equal to 0 is in $\mathcal{F}$. Then we can take $S_0 := \{f_0\}$ and $S_1 := \mathcal{F}$. We have for any $j \geq 1, S_j = \mathcal{F}$ and $p_j = \mathrm{Id}_{\mathcal{F}}$. The first assertion is then a direct application of Theorem 2.

From the first result of the corollary, for any $0 < \beta \leq 0.73$, with $\mathbb{P}^{2n}$-probability at least $1-\beta$, for any distribution ρ_n, we have

$$\rho_n P'_n f - \rho_n P_n f \leq 3.7 \sqrt{\tfrac{K(\rho_n, \pi) + \log(2\beta^{-1})}{n}} + \frac{3.7 \max_{[0;1]} \chi}{\sqrt{n}}.$$

The second assertion of the corollary then follows. ■

4.4 Data-dependent Bounds

The obtained bound is not completely empirical since it involves the expectation with respect to an extra sample. In the transduction setting, this is not an issue, it is even an advantage as one can use the unlabelled data in the computation of the bound. However, in the inductive setting, this is a drawback. Future work will focus on using concentration inequalities to give a fully empirical bound.

5. Conclusion and Perspectives

We have obtained a generalization error bound for randomized classifiers which combines several previous improvements. It contains an optimal union bound, both in the sense of optimally taking into account the metric structure of the set of functions (via the majorizing measure approach) and in the sense of taking into account the averaging distribution. It also is sensitive to the variance of the functions and is thus "localized".

In particular it is the first PAC-Bayesian bound that remains finite when the averaging distribution is concentrated at a point.

There still remains work in order to get a fully empirical bound and to better understand the connection with Rademacher averages. In particular, the way the approximating sets S_j should be constructed in practical cases has to be investigated.

Acknowledgments

We would like to thank Gilles Blanchard and the referees for their numerous useful comments and suggestions that have greatly improved the first version of this work.

Appendix A. Proof of Our Main Results

The proof of our main results is inspired by previous work on the PAC-Bayesian bounds (Catoni, 2003), Audibert (2004b) and on the generic chaining (Talagrand, 1996). Classical PAC-Bayesian bounds are mainly based on the combination of three ingredients:

1. the duality property of the entropy (Lemma 7),
2. Markov's inequality (which leads to Lemma 8),
3. and upper bounds on the exponential moment of a bounded random variable (which is the underlying idea of Lemma 9).

We reused these three main ingredients and slightly extended them to fit our needs. The additional extra step that is required to obtain the generic chaining aspect is to decompose the functions into a *chain* (as explained in Section 2.3.1) and to combine together all the individual PAC-Bayesian bounds obtained for each element of the chain. This chaining part is actually done in two steps: in the first one (Section A.1.4) we perform a non uniform union bound with appropriately chosen weights in order to make the bound independent on λ, while in the second one (Section A.1.5), we actually chain the inequalities to obtain the final result.

A.1 Proof of Theorem 2

In order to emphasize the various techniques that are used, we decompose its proof in a series of short steps.

A.1.1 LEGENDRE TRANSFORM OF THE KULLBACK-LEIBLER DIVERGENCE

The first step consists in using a duality property of the relative entropy (see, for example, Dembo and Zeitouni 1998, or page 10 of Catoni 2003). Namely, one has the following

Lemma 7 (Legendre transform of the KL-divergence) *For any π-measurable function $h : \mathcal{F} \to \mathbb{R}$ and any probability distribution ρ on $\mathcal{F}$,*

$$\rho h \leq \log \pi e^h + K(\rho, \pi).$$

Proof We have $\rho h - K(\rho,\pi) - \log \pi e^h = -K\big(\rho, \frac{e^h}{\pi e^h} \cdot \pi\big) \leq 0$ with equality when $\rho = \frac{e^h}{\pi e^h} \cdot \pi$. ■

Lemma 8 *Consider a function $h : \mathcal{F} \to \mathbb{R}$ and probability distributions π and ρ on $\mathcal{F}$ that depend on $Z_1, \ldots, Z_n, Z'_1, \ldots, Z'_n$ in a measurable way, we have for any $\beta > 0$, with probability at least $1 - \beta$ with respect to the samples distribution*

$$\rho h \leq \log \mathbb{E}^{2n} \pi e^h + K(\rho, \pi) + \log(\beta^{-1}).$$

Also, with probability at least $1-\beta$ *with respect to the first sample distribution,*

$$\mathbb{E}'^n \rho h \leq \log \mathbb{E}^{2n} \pi e^h + \mathbb{E}'^n K(\rho,\pi) + \log(\beta^{-1})\,.$$

Proof From Markov's inequality applied to the non-negative random variable πe^h, we obtain that for any $t>0$ $\mathbb{P}\left(\pi e^h > t\right) \leq \frac{1}{t}\mathbb{E}^{2n}\pi e^h$, hence for any $\beta>0$, with probability at least $1-\beta$ with respect to the samples distribution,

$$\log \pi e^h \leq \log \mathbb{E}^{2n} \pi e^h + \log(\beta^{-1})\,.$$

Then the proof of the first result follows from Lemma 7. The second assertion can be proved in a similar way (applying Markov's inequality conditionally to the second sample) and using the inequality $\mathbb{E}'^n \log \mathbb{E}^n .. \leq \log \mathbb{E}^{2n} ...$ ■

A.1.2 DEVIATION INEQUALITY

Lemma 9 *For any* $\lambda > 0$*, any function* $\mathcal{W}: \mathcal{F}\times\mathcal{Z}\to\mathbb{R}$ *and any exchangeable function* $\pi: \mathcal{X}^{2n} \to \mathcal{M}_+^1(\mathcal{F})$*, we have*

$$\mathbb{E}^{2n}\pi e^{\lambda P_n' \mathcal{W} - \lambda P_n \mathcal{W} - \frac{2\lambda^2}{n} P_{2n}\mathcal{W}^2} \leq 1.$$

Proof Denote $\Delta_i := \mathcal{W}(\cdot,Z_i') - \mathcal{W}(\cdot,Z_i)$ and

$$h := \lambda P_n' \mathcal{W} - \lambda P_n \mathcal{W} - \frac{2\lambda^2}{n} P_{2n}\mathcal{W}^2.$$

By the exchangeability of π, for any $\sigma \in \{-1;+1\}^n$, we have

$$\begin{aligned}\mathbb{E}^{2n}\pi e^h &= \mathbb{E}^{2n}\pi e^{-\frac{2\lambda^2}{n}P_{2n}\mathcal{W}^2 + \frac{\lambda}{n}\sum_{i=1}^n \Delta_i} \\ &= \mathbb{E}^{2n}\pi e^{-\frac{2\lambda^2}{n}P_{2n}\mathcal{W}^2 + \frac{\lambda}{n}\sum_{i=1}^n \sigma_i\Delta_i}.\end{aligned}$$

Now taking the expectation w.r.t. σ, where σ is a n-dimensional vector of Rademacher variables. We obtain

$$\begin{aligned}\mathbb{E}^{2n}\pi e^h &= \mathbb{E}^{2n}\pi \left[e^{-\frac{2\lambda^2}{n}P_{2n}\mathcal{W}^2} \prod_{i=1}^n \cosh\left(\frac{\lambda}{n}\Delta_i\right)\right] \\ &\leq \mathbb{E}^{2n}\pi \left[e^{-\frac{2\lambda^2}{n}P_{2n}\mathcal{W}^2} e^{\sum_{i=1}^n \frac{\lambda^2}{2n^2}\Delta_i^2}\right]\end{aligned}$$

where at the last step we use that $\cosh s \leq e^{\frac{s^2}{2}}$. The result follows from the inequality $\Delta_i^2 \leq 2\mathcal{W}^2(\cdot,Z_i') + 2\mathcal{W}^2(\cdot,Z_i)$. ■

A.1.3 CONSEQUENCES

We first prove the following lemma.

Lemma 10 *For any* $\beta > 0$*,* $\lambda > 0$*, any function* $\mathcal{W}: \mathcal{F}\times\mathcal{Z}\to\mathbb{R}$ *and any exchangeable function* $\pi: \mathcal{X}^{2n} \to \mathcal{M}_+^1(\mathcal{F})$*, with* $\mathbb{P}^n$*-probability at least* $1-\beta$*, for any probability distribution* $\rho_n \in \mathcal{M}_+^1(\mathcal{F})$*, we have*

$$\rho_n P_n' \mathcal{W} - \rho_n P_n \mathcal{W} \leq \frac{2\lambda}{n}\rho_n P_{2n}\mathcal{W}^2 + \frac{K(\rho_n,\pi)+\log(\beta^{-1})}{\lambda}.$$

Proof The result follows from Lemmas 8 and 9 applied to $h := \lambda P'_n \mathcal{W} - \lambda P_n \mathcal{W} - \frac{2\lambda^2}{n} P_{2n} \mathcal{W}^2$ and $\rho = \rho_n$. ■

Now let us apply this result to the projected measures $[\pi^{(j)}]_j$ and $[\rho_n]_j$ and for $\mathcal{W}(f,Z) = p_j(f)(Z) - p_{j-1}(f)(Z)$. Since, by definition, $\pi^{(j)}$, S_j and p_j are exchangeable, $[\pi^{(j)}]_j$ is also exchangeable. With $\mathbb{P}^{2n}$-probability at least $1-\beta$, uniformly in ρ_n, we have

$$[\rho_n]_j \Big\{ P'_n[p_j(f) - p_{j-1}(f)] - P_n[p_j(f) - p_{j-1}(f)] \Big\} \le \frac{2\lambda}{n} [\rho_n]_j d^2_{2n}[p_j(f), p_{j-1}(f)] + \frac{K'_j}{\lambda},$$

where $K'_j := K([\rho_n]_j, [\pi^{(j)}]_j) + \log(\beta^{-1})$. By definition of $[\rho_n]_j$, it implies that

$$\rho_n \Big\{ P'_n[p_j(f) - p_{j-1}(f)] - P_n[p_j(f) - p_{j-1}(f)] \Big\} \le \frac{2\lambda}{n} \rho_n d^2_{2n}[p_j(f), p_{j-1}(f)] + \frac{K'_j}{\lambda},$$

which, by using the notation introduced on page 874, can be shortened into

$$\rho_n \Delta_{n,j} \le \frac{2\lambda}{n} \rho_n d_j^2 + \frac{K'_j}{\lambda}, \tag{18}$$

The parameter $\lambda = \sqrt{\frac{nK'_j}{2\rho_n d_j^2}}$ minimizing the r.h.s. of the previous inequality depends on ρ_n. To use this data-dependent parameter, we need that (18) holds uniformly in λ.

A.1.4 WEIGHTED UNION BOUND ON THE PARAMETER λ

To get a uniform version of (18), introduce a grid $(\lambda_k)_{k \in \mathbb{N}^*}$ of $\mathbb{R}_+^*$. Let $(w_k)_{k \in \mathbb{N}^*}$ denote some positive real numbers such that $\sum_{k \ge 1} w_k = 1$. Define $\mathcal{B} := \inf_{k \ge 1} \left\{ \frac{2\rho_n d_j^2}{n} \lambda_k + \frac{K'_j + \log w_k^{-1}}{\lambda_k} \right\}$. Using a weighted union bound of (18), precisely using (18) for $(\lambda, \beta) = (\lambda_k, w_k \beta)$ for $k \in \mathbb{N}^*$, we get that, with probability at least $1-\beta$, we have $\rho_n \Delta_{n,j} \le \mathcal{B}$.

Our goal is then to choose the λ_k's and the w_k's such that $\mathcal{B}$ is the smallest possible. Ideally, we want to obtain a bound close to $a_\lambda := \min_{\lambda \in \mathbb{R}_+^*} \left\{ \frac{2\rho_n d_j^2}{n} \lambda + \frac{K'_j}{\lambda} \right\}$.

Let $m := e^{-\frac{1}{4}} \sqrt{\frac{n}{8}}$. Taking $\lambda_k = m e^{\frac{k}{2}}$ and $w_k = \frac{1}{k(k+1)}$, we are going to prove that we achieve this target up to a multiplicative constant and an additive $\log\log$ term.

First, since we have $\rho_n \Delta_{n,j} = 0$ when $\rho_n d_j^2 = 0$, we only focus on the case when $\rho_n d_j^2 > 0$. Define $\lambda^* := 2m \sqrt{\frac{K'_j}{\rho_n d_j^2}}$. We have $\rho_n d_j^2 \le 4$ and $K'_j \ge 1$ for $\beta \le e^{-1}$, hence $\lambda^* \ge m$. So there exists $k^* \in \mathbb{N}^*$ such that $\lambda_{k^*} e^{-\frac{1}{2}} \le \lambda^* < \lambda_{k^*}$. We have

$$\begin{aligned}
\mathcal{B} &\le \frac{2\rho_n d_j^2}{n} \lambda_{k^*} + \frac{K'_j + \log w_{k^*}^{-1}}{\lambda_{k^*}} \\
&\le \frac{2e^{\frac{1}{2}} \rho_n d_j^2}{n} \lambda^* + \frac{K'_j + \log[k^*(k^*+1)]}{\lambda^*} \\
&\le 2\sqrt{2} e^{\frac{1}{4}} \sqrt{\frac{\rho_n d_j^2 K'_j}{n}} + \frac{\log[k^*(k^*+1)]}{\lambda^*}.
\end{aligned}$$

The inequality $\lambda_{k^*}e^{-\frac{1}{2}} \leq \lambda^*$ implies $k^* - 1 \leq 2\log\left(\frac{\lambda^*}{m}\right)$, hence $k^*+1 \leq \log\left(4e^2\frac{K'_j}{\rho_n d_j^2}\right)$. Finally we have proved that with probability at least $1-\beta$,

$$\rho_n\Delta_{n,j} \leq 2\sqrt{2}e^{\frac{1}{4}}\sqrt{\frac{\rho_n d_j^2 K'_j}{n}} + \frac{2\sqrt{2}e^{\frac{1}{4}}}{\sqrt{n}}\chi\left(\frac{\rho_n d_j^2}{K'_j}\right).$$

We recall that $K'_j := K([\rho_n]_j, [\pi^{(j)}]_j) + \log(\beta^{-1})$ and $\chi(x) := \sqrt{x}\log\log\left(4e^2/x\right)$.

A.1.5 Chaining the Inequalities

By simply using a union bound with weights equal to $\frac{1}{j(j+1)}$, the previous inequality holds[4] uniformly in $j \in \mathbb{N}^*$ provided that β is replaced with $\beta/[j(j+1)]$, hence to apply the result of the previous section we need that $\beta/2 \leq e^{-1}$.

Since $p_{j-1} = p_{j-1} \circ p_j$, we have

$$\begin{aligned}\rho_n\left[P'_n f - P'_n f_0 + P_n f_0 - P_n f\right] \\ &= \rho_n\Delta_{n,J}(f) + \rho_n\left\{\textstyle\sum_{j=1}^J \left[(P'_n - P_n)p_j(f) - (P'_n - P_n)p_{j-1}(f)\right]\right\} \\ &= \rho_n\Delta_{n,J}(f) + \textstyle\sum_{j=1}^J \rho_n\left[(P'_n - P_n)p_j(f) - (P'_n - P_n)p_{j-1}(f)\right] \\ &= \rho_n\Delta_{n,J}(f) + \textstyle\sum_{j=1}^J [\rho_n]_j\left[(P'_n - P_n)f - (P'_n - P_n)p_{j-1}(f)\right]\end{aligned}$$

Setting $K_j := K([\rho_n]_j, [\pi^{(j)}]_j) + \log[j(j+1)\beta^{-1}]$, with $\mathbb{P}^n$-probability at least $1-\beta$, for any distribution ρ_n, we have

$$\begin{aligned}\rho_n\left[P'_n f - P'_n f_0 + P_n f_0 - P_n f\right] \leq \sup_{\mathcal{F}}\Delta_{n,J} + 2\sqrt{2}e^{\frac{1}{4}}\textstyle\sum_{j=1}^J\sqrt{\frac{\rho_n d_j^2 K j}{n}} \\ + \frac{2\sqrt{2}e^{\frac{1}{4}}}{\sqrt{n}}\textstyle\sum_{j=1}^J\chi\left(\frac{\rho_n d_j^2}{K_j}\right).\end{aligned}$$

Making $J \to +\infty$, we obtain Theorem 2.

A.2 Proof of Theorem 3

It suffices to modify Lemma 10 in the proof of Theorem 2. Indeed, using the second part of Lemma 8 instead of the first one we get

$$\rho_n P\mathcal{W} - \rho_n P_n\mathcal{W} \leq \frac{2\lambda}{n}\mathbb{E}'^n\rho_n P_{2n}\mathcal{W}^2 + \frac{\mathbb{E}'^n K(\rho_n,\pi) + \log(\beta^{-1})}{\lambda}.$$

The remaining parts of the proof (i.e., the union bound and the chaining) are similar.

Appendix B. Additional Material

Lemma 11 *For any random variable Z_f which is $\pi\otimes P$ measurable we have*

$$\log\pi e^{\mathbb{E}[Z_f]} \leq \mathbb{E}\left[\log\pi e^{Z_f}\right] \leq \log\pi\mathbb{E}\left[e^{Z_f}\right].$$

4. This is because $\sum_{j\in\mathbb{N}^*}\frac{1}{j(j+1)} = 1$.

Proof By duality (Lemma 7), we have

$$\mathbb{E}\left[\log \pi e^{Z_f}\right] = \mathbb{E}\left[\sup_{\rho_n}\left\{\rho_n Z_f - K(\rho_n,\pi)\right\}\right] \geq \sup_{\rho_n} \rho_n \mathbb{E}\left[Z_f\right] - K(\rho_n,\pi) = \log \pi e^{\mathbb{E}\left[Z_f\right]}.$$

This gives the first inequality. The second inequality follows from Jensen's inequality (applied to the convex function $-\log$) and Fubini's theorem. ■

B.1 Concentration Inequalities

In this section, we recall some concentration inequalities whose proofs can be found in Lugosi (2003). We start with Markov's inequality

Theorem 12 (Markov's inequality) *For any real-valued random variable X,*

$$\mathbb{P}(X >= t) \leq e^{-t}\mathbb{E}\left[e^X\right].$$

Theorem 13 (Hoeffding's inequality, 1963) *For any centered random variable X such that $a \leq X \leq b$, and any $\lambda > 0$, we have $\mathbb{E}e^{\lambda X} \leq e^{\frac{\lambda^2(b-a)^2}{8}}$. As a consequence, for any i.i.d. random variables $X_1,\dots,X_n$ such that $a \leq X_i \leq b$, we have*

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^n X_i > t\right) \leq e^{-\frac{2nt^2}{(b-a)^2}}.$$

Theorem 14 (Bernstein's inequality) *Let $X_1,\dots,X_n$ be n i.i.d. centered random variables such that $a \leq X_i \leq b$. We have*

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^n X_i > t\right) \leq e^{-\frac{nt^2}{2\operatorname{Var}X + \frac{2(b-a)t}{3}}}.$$

We say that a function has the bounded differences property if for some constant $c > 0$

$$\sup_{x_1,\dots,x_n,x' \in \mathcal{X}, i \in \{1,\dots,n\}} \left|g(x_1,\dots,x_n) - g(x_1,\dots,x_{i-1},x',x_{i+1},\dots,x_n)\right| \leq c.$$

Theorem 15 (McDiarmid's inequality) *Let g satisfy the bounded differences assumption with constant c. Then we have*

$$\mathbb{P}\left[g(X_1,\dots,X_n) - \mathbb{E}g(X_1,\dots,X_n) \geq t\right] \leq e^{-\frac{2t^2}{nc^2}}.$$

Note that McDiarmid's result generalizes Hoeffding's inequality.

B.2 Symmetrization Inequalities

Lemma 16 (Symmetrization in probability, Vapnik and Chervonenkis 1971) *Assume the functions in $\mathcal{F}$ have range in $[a;b]$. For any $t > 0$ such that $nt^2 \geq 2(b-a)^2$,*

$$\mathbb{P}^n\left(\sup_{f\in\mathcal{F}}\left\{Pf - P_nf\right\} \geq t\right) \leq 2\mathbb{P}^{2n}\left(\sup_{f\in\mathcal{F}}\left\{P_n'f - P_nf\right\} \geq \frac{t}{2}\right).$$

Proof Let f_n be the function (depending on the first sample) achieving the supremum of $(P-P_n)f$ over $\mathcal{F}$ (if it does not exist, one can use a limiting argument). We have $\mathbb{1}_{(P-P_n)f_n>t}\mathbb{1}_{(P-P'_n)f_n<\frac{t}{2}} \le \mathbb{1}_{(P'_n-P_n)f>\frac{t}{2}}$. Taking expectations w.r.t. the second sample gives $\mathbb{1}_{(P-P_n)f_n>t}\mathbb{P}'^n[(P-P'_n)f_n<\frac{t}{2}] \le \mathbb{P}'^n[(P'_n-P_n)f>\frac{t}{2}]$. Now by Chebyshev's inequality, we have $\mathbb{P}'^n[(P-P'_n)f_n \ge \frac{t}{2}] \le \frac{4\operatorname{Var} f_n}{nt^2} \le \frac{(b-a)^2}{nt^2} \le \frac{1}{2}$. We obtain $\mathbb{1}_{(P-P_n)f_n>t} \le 2\mathbb{P}'^n[(P'_n-P_n)f>\frac{t}{2}]$ and conclude by taking expectations w.r.t. the first sample. ■

Remark 6 *By replacing Chebyshev's inequality with Bernstein's inequality, we can improve the previous result to take into account t of order smaller than $1/\sqrt{n}$. One can also slightly generalize the previous result to obtain: for any positive reals η and t,*

$$\mathbb{P}^n\left(\sup_{f\in\mathcal{F}}\{Pf-P_nf\}\ge t\right) \le \frac{\mathbb{P}^{2n}\left(\sup_{f\in\mathcal{F}}\{P'_nf-P_nf\}\ge(1-\eta)t\right)}{1-e^{-\frac{n\eta^2t^2}{2\operatorname{Var}f+2(b-a)\eta t/3}}}.$$

Lemma 17 (Symmetrization for expectations, Giné and Zinn 1984) *For any set $\mathcal{F}$ of functions,*

$$\mathbb{E}^n\sup_{f\in\mathcal{F}}\{Pf-P_nf\} \le \frac{2}{n}\mathbb{E}^n\mathbb{E}_\sigma\sup_{f\in\mathcal{F}}\sum_{i=1}^n\sigma_if(Z_i).$$

Proof We have

$$\begin{aligned}\mathbb{E}^n\sup_{f\in\mathcal{F}}\{Pf-P_nf\} &= \mathbb{E}^n\sup_{f\in\mathcal{F}}\{\mathbb{E}'^n[P'_nf]-P_nf\}\\ &\le \mathbb{E}^{2n}\sup_{f\in\mathcal{F}}\{P'_nf-P_nf\}\\ &= \mathbb{E}^{2n}\mathbb{E}_\sigma\sup_{f\in\mathcal{F}}\tfrac{1}{n}\sum\sigma_i(f(Z'_i)-f(Z_i))\\ &\le \tfrac{2}{n}\mathbb{E}^n\mathbb{E}_\sigma\sup_{f\in\mathcal{F}}\sum\sigma_if(Z_i).\end{aligned}$$

■

The following lemmas, due to Panchenko, allow to convert transductive bounds into inductive ones.

Lemma 18 *For any function $B:\mathcal{Z}^{2n}\to\mathbb{R}$, if for any $\beta>0$, with $\mathbb{P}^{2n}$-probability at least $1-\beta$, we have $B\le\log(\beta^{-1})$, then for any $\beta>0$, with $\mathbb{P}^n$-probability $1-e\beta$, we have*

$$\mathbb{E}'^nB\le\log(\beta^{-1}).$$

Proof It directly comes from Lemma 1 in Panchenko (2003). ■

Lemma 19 *Let B_1, B_2 and B_3 be three functions of $Z_1,\dots,Z_n$ and $Z'_1,\dots,Z'_n$ with $B_2\ge0$ and $B_3\ge0$. If for any $\beta>0$, with $\mathbb{P}^{2n}$-probability at least $1-\beta$, we have*

$$B_1\le\sqrt{B_2(B_3+\log(\beta^{-1}))}$$

then for all $\beta > 0$, *with* $\mathbb{P}^n$*-probability* $1 - e\beta$,

$$\mathbb{E}'^n B_1 \leq \sqrt{\mathbb{E}'^n B_2 \left[\mathbb{E}'^n B_3 + \log(\beta^{-1})\right]}.$$

Proof It suffices to modify slightly the proof of Corollary 1 in Panchenko (2003). Specifically, we apply Lemma 18 to the quantity $B := \sup_{\lambda>0}\left\{4\lambda(B_1 - \lambda B_3) - B_2\right\}$ since simple computations show that $\left\{B \geq \log(\beta^{-1})\right\} = \left\{B_1 \geq \sqrt{B_3(B_2 + \log(\beta^{-1}))}\right\}$.

■

B.3 Proof of Known Results

Here we prove the results presented in the survey section (see Section 2).

B.3.1 Proof of Inequality (5)

For any $t \in \mathbb{R}$, Hoeffding's inequality (see Section B.1) implies $\mathbb{P}^n\left[Pf - P_n f > t\right] \leq e^{-2nt^2}$. Choosing $\beta = e^{-2nt^2}$, we obtain

$$Pf - P_n f \leq \frac{1}{\sqrt{2}}\sqrt{\frac{\log(\beta^{-1})}{n}}.$$

B.3.2 Proof of Inequality (6)

Statement (5) holds uniformly over $|\mathcal{F}|$ functions with probability at least $1 - |\mathcal{F}|\beta$. Setting $\beta' = |\mathcal{F}|\beta$, we obtain the desired result.

B.3.3 Proof of Inequality (7) (Vapnik and Chervonenkis, 1971)

First, use the symmetrization lemma 16. Denote $\mathcal{F}_{Z,Z'}$ the set of vectors formed by the values of each function in $\mathcal{F}$ on the double sample. Let σ_i be independent random signs ($+1, -1$ with probability $1/2$). We have

$$\begin{aligned}
\mathbb{P}^{2n}\left(\sup_{f\in\mathcal{F}}\left\{P'_n f - P_n f\right\} \geq \tfrac{t}{2}\right) &= \mathbb{P}^{2n}\left(\sup_{f\in\mathcal{F}_{Z,Z'}}\left\{P'_n f - P_n f\right\} \geq \tfrac{t}{2}\right)\\
&= \mathbb{P}_\sigma \mathbb{P}^{2n}\left(\sup_{f\in\mathcal{F}_{Z,Z'}} \tfrac{1}{n}\textstyle\sum_{i=1}^n \sigma_i[f(Z_i) - f(Z'_i)] \geq \tfrac{t}{2}\right)\\
&= \mathbb{E}^{2n}\mathbb{E}_\sigma \sup_{f\in\mathcal{F}_{Z,Z'}} \mathbb{1}_{\frac{1}{n}\sum_{i=1}^n \sigma_i[f(Z_i)-f(Z'_i)]\geq\frac{t}{2}}\\
&\leq \mathbb{E}^{2n}\mathbb{E}_\sigma \sum_{f\in\mathcal{F}_{Z,Z'}} \mathbb{1}_{\frac{1}{n}\sum_{i=1}^n \sigma_i[f(Z_i)-f(Z'_i)]\geq\frac{t}{2}}\\
&\leq \mathbb{E}^{2n}|\mathcal{F}_{Z,Z'}| \sup_{b\in\{-1;0;+1\}^n} \mathbb{P}_\sigma\left(\tfrac{1}{n}\textstyle\sum_{i=1}^n b_i\sigma_i \geq \tfrac{t}{2}\right)\\
&\leq e^{-\frac{nt^2}{8}}\mathbb{E}^{2n}|\mathcal{F}_{Z,Z'}|,
\end{aligned}$$

where the last step comes from Hoeffding's inequality. Putting $\beta = 2\mathbb{E}^{2n}|\mathcal{F}_{Z,Z'}|e^{-\frac{nt^2}{8}}$, we get with probability at least $1-\beta$,

$$\forall f \in \mathcal{F},\, Pf - P_n f \leq \sqrt{8}\sqrt{\frac{\log \mathbb{P}^{2n} N(\mathcal{F}, 1/n, d_{2n}) + \log(2\beta^{-1})}{n}}.$$

B.3.4 PROOF OF INEQUALITY (13) (MCALLESTER, 1998)

We use the same C as for statement (5). The probability that (13) does not hold is upper bounded with

$$\sum_{f\in\mathcal{F}} \mathbb{P}^n\left[Pf - P_nf > C\sqrt{\frac{\log 1/[\beta\pi(f)]}{n}}\right] \leq \sum_{f\in\mathcal{F}} \pi(f)\beta = \beta.$$

B.3.5 PROOF OF INEQUALITY (11)

To prove (11), we replace Hoeffding's inequality with Bernstein's inequality (see Section B.1): $\mathbb{P}^n\left[Pf - P_nf > t\right] \leq e^{-\frac{nt^2}{2\mathrm{Var} f + 2t/3}}$. Then choosing β equal to the right-hand side gives the result.

B.3.6 PROOF OF INEQUALITY (14) (MCALLESTER, 1999; SEEGER, 2003)

Let $c \geq 1/2$. Here we propose a slightly different proof in order to get rid of the logarithmic term in (14). Using Jensen's inequality and Lemma 7, we get

$$[\rho_n(Pf - P_nf)_+]^2 \leq \rho_n(Pf - P_nf)_+^2 \leq \frac{c}{n}\left(\log \pi e^{\frac{n}{c}(Pf-P_nf)_+^2} + K(\rho_n,\pi)\right).$$

Moreover, by Markov's inequality and Fubini's Theorem, we have

$$\mathbb{P}^n\left[\log \pi e^{\frac{n}{c}(Pf-P_nf)_+^2} > t\right] \leq e^{-t}\pi\mathbb{E}^n\left[e^{\frac{n}{c}(Pf-P_nf)_+^2}\right],$$

and

$$\begin{array}{rcl} \mathbb{E}^n e^{\frac{n}{c}(Pf-P_nf)_+^2} & = & \int_0^{+\infty} \mathbb{P}^n\left(e^{\frac{n}{c}(Pf-P_nf)_+^2} \geq u\right)du \\ & = & 1 + \int_1^{e^{n/c}} \mathbb{P}^n\left(Pf - P_nf \geq \sqrt{\frac{c\log u}{n}}\right)du \\ & \leq & 1 + \int_1^{e^{n/c}} \frac{du}{u^{-2c}} \\ & = & 1 + \frac{1-e^{-n(2c-1)/c}}{2c-1}, \end{array}$$

where the inequality comes from the proof of (5). Taking $\beta = 1 + \frac{1-e^{-n(2c-1)/c}}{2c-1}e^{-t}$, we obtain for $c = 1/2$, with probability at least $1-\beta$,

$$\rho_n(Pf - P_nf) \leq \frac{1}{\sqrt{2}}\sqrt{\frac{K(\rho_n,\pi) + \log(2n+1) + \log 1/\beta}{n}}$$

and for any $c > \frac{1}{2}$, with probability at least $1-\beta$,

$$\rho_n(Pf - P_nf) \leq \sqrt{c}\sqrt{\frac{K(\rho_n,\pi) + \log(\frac{2c}{2c-1}) + \log 1/\beta}{n}}.$$

B.3.7 PROOF OF INEQUALITY (10)

From Theorem 15 and Lemma 17, we obtain that with probability at least $1-\beta$,

$$\sup_{f\in\mathcal{F}}(Pf - P_nf) \leq \frac{2}{n}\mathbb{E}^n\left[\mathbb{E}_\sigma \sup_{f\in\mathcal{F}} \sum \sigma_i f(Z_i)\right] + \frac{1}{\sqrt{2}}\sqrt{\frac{\log(\beta^{-1})}{n}}. \tag{19}$$

B.3.8 PROOF OF INEQUALITY (8)

The starting point is Inequality (19). Let f_n be the function achieving the supremum. (To shorten the proof, we assume its existence.) Introduce the vectors h^* and $h^{(0)}$ such that $h_i^* = f_n(Z_i)$ and $h_i^{(0)} = 0$. Consider the canonical distance on $\mathbb{R}^n$: $\|x\| := \sqrt{\sum x_i^2}$, and take the minimal covering nets $\mathcal{N}_k$ of the set of vectors $\{[f(Z_i)]_{i=1,\dots,n} : f \in \mathcal{F}\}$ of respective radius 2^{-k} for $k = 1, \dots, K$ where $K := \lfloor \log_2 \sqrt{n} \rfloor + 1$. Let $h^{(k)}$ be a nearest neighbour of h^* in the net $\mathcal{N}_k$. Let $e_k := \mathbb{E}_\sigma \max_{(h',h'') \in \bar{\bar{\mathcal{N}}}_k} \sum_i \sigma_i (f_i - g_i)$ and

$$\bar{\bar{\mathcal{N}}}_k := \left\{ h' \in \mathcal{N}_k, h'' \in \mathcal{N}_{k-1} : \|h' - h''\| \leq 32^{-k} \right\}.$$

We have

$$\begin{aligned}
\mathbb{E}_\sigma \sup_{f \in \mathcal{F}} \sum \sigma_i f(Z_i) &= \mathbb{E}_\sigma \sum \sigma_i h_i^* \\
&= \mathbb{E}_\sigma \sum \sigma_i \left(h_i^* - h_i^{(K)} \right) + \mathbb{E}_\sigma \sum_{i,k} \sigma_i \left(h_i^{(k)} - h_i^{(k-1)} \right) \\
&\leq 1 + \mathbb{E}_\sigma \sum_{i,k} \sigma_i \left(h_i^{(k)} - h_i^{(k-1)} \right) \\
&\leq 1 + \sum_k \mathbb{E}_\sigma \sum_i \sigma_i \left(h_i^{(k)} - h_i^{(k-1)} \right) \\
&\leq 1 + \sum_k e_k.
\end{aligned}$$

From the following lemma, using that for any $(h', h'') \in \bar{\bar{\mathcal{N}}}_k$, $\|h' - h''\| \leq 3 \times 2^{-k}$ and $|\bar{\bar{\mathcal{N}}}_k| \leq |\mathcal{N}_k|^2$, we get $e_k \leq 3 \times 2^{-k} \sqrt{4n \log N(\mathcal{F}, 2^{-k}, d_n)}$.

Lemma 20 (Pisier 1986) *Let $\sigma_i, i = 1, \dots, n$ be Rademacher variables and $c_j, j = 1, \dots, J$ be $\mathbb{R}^n$-vectors such that for any $j \in \{1, \dots, J\}$, $\|c_j\| \leq c$. Then we have*

$$\mathbb{E} \max_{j \in \{1,\dots,J\}} \sum_i \sigma_i c_{j,i} \leq c \sqrt{2n \log J}.$$

Proof For any $\lambda > 0$, we have

$$\mathbb{E} \max_{j \in \{1,\dots,J\}} \sum_i \sigma_i c_{j,i} \leq \frac{1}{\lambda} \log \sum_j \mathbb{E} e^{\sum_i \lambda \sigma_i c_{j,i}} \leq \frac{\log J}{\lambda} + \frac{\lambda n c^2}{2}.$$

Optimizing the parameter λ, the upper bound becomes $c\sqrt{2n \log J}$. ■

Then we have

$$\begin{aligned}
\textstyle\sum_k e_k &\leq 12\sqrt{n} \textstyle\sum 2^{-(k+1)} \sqrt{\log N(\mathcal{F}, 2^{-k}, d_n)} \\
&\leq 12\sqrt{n} \textstyle\int_0^1 \sqrt{\log N(\mathcal{F}, r, d_n)} dr.
\end{aligned}$$

To conclude, the chaining trick showed that Rademacher averages are bounded by the Koltchinskii-Pollard integral.

B.3.9 PROOF OF INEQUALITY (9)

The proof is inspired from Talagrand (1996). Let $\left(\pi^{(j)}\right)_{j\in\mathbb{N}}$ be a family of probability distributions on $\mathcal{F}$. We want to prove that the Rademacher averages are bounded with $C\sup_{f\in\mathcal{F}}\sum_{j=1}^{\infty} r^{-j}\sqrt{\log\{1/\pi^{(j)}[A_j(f)]\}}$. For any $j\in\mathbb{N}$ and any $A\in\mathcal{A}_j$, we choose an arbitrary point $x_A\in A$. Let us define $p_j(f):=x_{A_j(f)}$ and $S_j:=\{p_j(f):f\in\mathcal{F}\}$. We have $\mathcal{A}_0=\{\mathcal{F}\}$. Let $f_0=x_{\mathcal{F}}$. Define $X_f:=\sum_i\sigma_i f(Z_i)$.

From Cauchy-Schwarz inequality, we have $\sup_{\sigma,Z_1,\dots,Z_n,f}|X_f-X_{p_j(f)}|\le\sqrt{n}r^{-j}$, hence $\sum_{j\ge1}[X_{p_j(f)}-X_{p_{j-1}(f)}]$ converges uniformly towards $X_f-X_{f_0}$. Introduce a probability distribution π' such that $\pi'(\{x_A\})\ge 2^{-j-1}\pi^{(j)}(A)$ for any $A\in\mathcal{A}_j$. Define the quantities $a_j(f):=r^{-j+1}\sqrt{2n\log[2/\pi'(f)]}$ and $M:=\sup_{f\in\mathcal{F}}\sum_{j\ge1}a_j[p_j(f)]$. We have

$$\begin{array}{rcl}\mathbb{E}_\sigma\exp\left\{\lambda(X_f-X_g)\right\} & = & \prod_{i=1}^n\mathbb{E}_\sigma\exp\left\{\sigma_i\lambda[f(Z_i)-g(Z_i)]\right\}\\ & = & \prod_{i=1}^n\cosh\{\lambda[f(Z_i)-g(Z_i)]\}\\ & \le & \prod_{i=1}^n\exp\left\{\frac{\lambda^2[f(Z_i)-g(Z_i)]^2}{2}\right\}\\ & = & e^{\frac{n\lambda^2 d_n^2(f,g)}{2}},\end{array}$$

hence for any $u>0$, $\mathbb{P}_\sigma(X_f-X_g\ge u)\le e^{-\frac{u^2}{2nd_n^2(f,g)}}$. Then for any $u\ge1$, we get

$$\begin{array}{rcl}\mathbb{P}_\sigma\left(\sup_{f\in\mathcal{F}}\left\{X_f-X_{f_0}\right\}\ge uM\right) & \le & \sum_{j\ge1,v\in S_j}\mathbb{P}_\sigma\left[X_v-X_{p_{j-1}(v)}\ge ua_j(v)\right]\\ & \le & \sum_{j\ge1,v\in S_j}e^{-\frac{u^2a_j^2(v)}{2n\mathrm{Diam}^2A_{j-1}(v)}}\\ & \le & \sum_{j\ge1,v\in S_j}e^{-u^2\log\frac{2}{\pi'(v)}}\\ & \le & 2^{1-u^2},\end{array}$$

since $\pi'(v)^{u^2}\le\pi'(v)$. We obtain

$$\mathbb{E}_\sigma\sup_{f\in\mathcal{F}}X_f=\mathbb{E}_\sigma\sup_{f\in\mathcal{F}}\left\{X_f-X_{f_0}\right\}\le\int_0^{+\infty}2^{1-u^2}du M\le 2.2M.$$

By plugging the definitions of X_f and M, we obtain

$$\mathbb{E}_\sigma\sup_{f\in\mathcal{F}}\sum_i\sigma_i f(Z_i)\le 4\sqrt{n}\sup_{f\in\mathcal{F}}\sum_{j\ge1}r^{-j+1}\sqrt{\log\{2^{j+2}/\pi^{(j)}[A_j(f)]\}}.$$

From (19) and by using that $\sqrt{\log\{2^{j+2}/\pi^{(j)}[A_j(f)]\}}\le\sqrt{j+2}+\sqrt{\log\{1/\pi^{(j)}[A_j(f)]\}}$, we get the desired result.

References

J.-Y. Audibert. Aggregated estimators and empirical complexity for least square regression. *Ann. Inst. Henri Poincaré, Probab. Stat.*, 40(6):685–736, 2004a.

J.-Y. Audibert. A better variance control for PAC-Bayesian classification. Technical report n.905, `http://www.proba.jussieu.fr/mathdoc/textes/PMA-905Bis.pdf`, Laboratoire de Probabilités et Modèles Aléatoires, Universités Paris 6 and Paris 7, 2004b.

P. L. Bartlett, O. Bousquet, and S. Mendelson. Local Rademacher complexities. *Ann. Stat.*, 33(4): 1497–1537, 2005.

P. L. Bartlett and S. Mendelson. Empirical minimization. *Probability Theory and Related Fields*, 135(3):311 – 334, 2006.

P. L. Bartlett, S. Mendelson, and P. Philips. Local complexities for empirical risk minimization. In J. Shawe-Taylor, editor, *17th Annual Conference on Learning Theory, COLT 2004*, LNCS-3120, berlin, 2004. Springer-Verlag.

S. Boucheron, O. Bousquet, and G. Lugosi. Theory of classification: a survey of some recent advances. *ESAIM: Probability and Statistics*, 9:323–375, 2005.

S. Boucheron, G. Lugosi, and S. Massart. A sharp concentration inequality with applications. *Random Structures and Algorithms*, 16:277–292, 2000.

O. Catoni. A PAC-Bayesian approach to adaptive classification. Technical report n.840, http://www.proba.jussieu.fr/mathdoc/preprints/, Laboratoire de Probabilités et Modèles Aléatoires, Universités Paris 6 and Paris 7, 2003.

A. Dembo and O. Zeitouni. *Large Deviation Techniques and Applications*. Springer, 1998.

L. Devroye and G. Lugosi. *Combinatorial Methods in Density Estimation*. Springer Series in Statistics. Springer Verlag, New York, 2001.

R. M. Dudley. A course on empirical processes. *Lecture Notes in Mathematics*, 1097:2–142, 1984.

E. Giné and J. Zinn. Some limit theorems for empirical processes. *Ann. Probab.*, 12(4):929–989, 1984.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

V. Koltchinskii. Local Rademacher complexities and oracle inequalities in risk minimization. *Annals of Statistics*, 34(6), 2006.

G. Lugosi. Concentration of measure inequalities. *Lecture notes*, pages 1–62, 2003. available from http://www.econ.upf.es/ lugosi/anu.ps.

P. Massart. Some applications of concentration inequalities to statistics. *Ann. Fac. Sci. Toulouse*, Math. 9(2):245–303, 2000.

P. Massart. *Concentration Inequalities and Model Selection: Ecole d'été de Probabilités de Saint-Flour XXXIII - 2003*. Lecture Notes in Mathematics. Springer, 2006.

D. A. McAllester. Some PAC-Bayesian theorems. In *Proceedings of the 11th Annual Conference on Computational Learning Theory*, pages 230–234. ACM Press, 1998.

D. A. McAllester. PAC-Bayesian model averaging. In *Proceedings of the 12th Annual Conference on Computational Learning Theory*. ACM Press, 1999.

D. Panchenko. Symmetrization approach to concentration inequalities for empirical processes. *Annals of Probability*, 31(4):2068–2081, 2003.

G. Pisier. Probabilistic methods in the geometry of banach spaces. In *Probability and analysis, Lect. Sess. C.I.M.E Varena, Italy 1985, Lecture Notes in Mathematics*, volume 1206, pages 167–241, Berlin, 1986. Springer.

M. Seeger. Bayesian gaussian process models: PAC-bayesian generalisation error bounds and sparse approximations. *PhD Thesis, University of Edinburgh*, December 2003.

M. Talagrand. Majorizing measures: The generic chaining. *Annals of Probability*, 24(3):1049–1103, 1996.

M. Talagrand. Majorizing measures without measures. *Annals of Probability*, 29(1):411–417, 2001.

M. Talagrand. *The Generic Chaining: Upper and Lower Bounds for Stochastic Processes*. Springer, 2005.

A. B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *Ann. Stat.*, 32(1):135–166, 2004.

S. van de Geer. *Empirical Processes in M-Estimation*. Cambridge University Press, Cambridge, UK, 2000.

A. W. van der Vaart. *Asymptotic Statistics*. Cambridge University Press, 1998.

A. W. van der Vaart and J. A. Wellner. *Weak Convergence and Empirical Processes*. Springer, 1996.

V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

V. Vapnik and A. Chervonenkis. *Theory of Pattern Recognition [in Russian]*. Nauka, Moscow, 1974. (German Translation: W. Wapnik & A. Tscherwonenkis, *Theorie der Zeichenerkennung*, Akademie–Verlag, Berlin, 1979).

Journal of Machine Learning Research 8 (2007) 891-933 Submitted 1/06; Revised 12/06; Published 5/07

Anytime Learning of Decision Trees

Saher Esmeir ESAHER@CS.TECHNION.AC.IL
Shaul Markovitch SHAULM@CS.TECHNION.AC.IL
Department of Computer Science
Technion—Israel Institute of Technology
Haifa 32000, Israel

Editor: Claude Sammut

Abstract

The majority of existing algorithms for learning decision trees are greedy—a tree is induced top-down, making locally optimal decisions at each node. In most cases, however, the constructed tree is not globally optimal. Even the few non-greedy learners cannot learn good trees when the concept is difficult. Furthermore, they require a fixed amount of time and are not able to generate a better tree if additional time is available. We introduce a framework for anytime induction of decision trees that overcomes these problems by trading computation speed for better tree quality. Our proposed family of algorithms employs a novel strategy for evaluating candidate splits. A biased sampling of the space of consistent trees rooted at an attribute is used to estimate the size of the minimal tree under that attribute, and an attribute with the smallest expected tree is selected. We present two types of anytime induction algorithms: a *contract* algorithm that determines the sample size on the basis of a pre-given allocation of time, and an *interruptible* algorithm that starts with a greedy tree and continuously improves subtrees by additional sampling. Experimental results indicate that, for several hard concepts, our proposed approach exhibits good anytime behavior and yields significantly better decision trees when more time is available.

Keywords: anytime algorithms, decision tree induction, lookahead, hard concepts, resource-bounded reasoning

1. Introduction

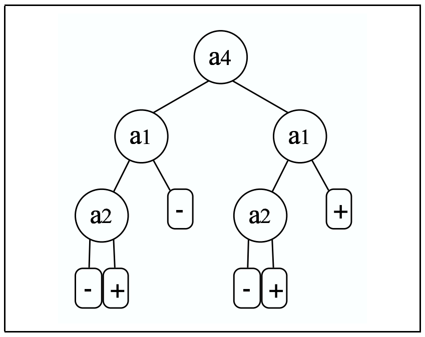
Assume that a medical center has decided to use medical records of previous patients in order to build an automatic diagnostic system for a particular disease. The center applies the C4.5 algorithm on thousands of records, and after few seconds receives a decision tree. During the coming months, or even years, the same induced decision tree will be used to predict whether patients have or do not have the disease. Obviously, the medical center is willing to wait much longer to obtain a better tree—either more accurate or more comprehensible.

Consider also a planning agent that has to learn a decision tree from a given set of examples, while the time at which the model will be needed by the agent is not known in advance. In this case, the agent would like the learning procedure to learn the best tree it can until it is interrupted and queried for a solution.

In both of the above scenarios, the learning algorithm is expected to exploit additional time allocation to produce a better tree. In the first case, the additional time is allocated in advance. In the second, it is not. Similar resource-bounded reasoning situations may occur in many real-life

a_1	a_2	a_3	a_4	label
1	0	0	1	+
0	1	0	0	+
0	0	0	0	-
1	1	0	0	-
0	1	1	1	+
0	0	1	1	-
1	0	1	1	+

(a) A set of training instances.



(b) ID3's performance.

Figure 1: Learning the 2-XOR concept $a_1 \oplus a_2$, where a_3 and a_4 are irrelevant

applications such as game playing, planning, stock trading and e-mail filtering. In this work, we introduce a framework for exploiting extra time, preallocated or not, in order to learn better models.

Despite the recent progress in advanced induction algorithms such as SVM (Vapnik, 1995), *decision trees* are still considered attractive for many real-life applications, mostly due to their interpretability (Hastie et al., 2001, chap. 9). Craven (1996) lists several reasons why the understandability of a model by humans is an important criterion for evaluating it. These reasons include, among others, the possibility for human validation of the model and generation of human-readable explanations for the classifier predictions. When classification cost is important, decision trees may be attractive in that they ask only for the values of the features along a single path from the root to a leaf. In terms of accuracy, decision trees have been shown to be competitive with other classifiers for several learning tasks.

The majority of existing methods for decision tree induction build a tree top-down and use local measures in an attempt to produce small trees, which, by Occam's Razor (Blumer et al., 1987), should have better predictive power. The well-known C4.5 algorithm (Quinlan, 1993) uses the gain ratio as a heuristic for predicting which attribute will yield a smaller tree. Several other alternative local greedy measures have been developed, among which are ID3's information gain, Gini index (Breiman et al., 1984), and chi-square (Mingers, 1989). Mingers (1989) reports an empirical comparison of several measures, and concludes that the predictive accuracy of the induced trees is not sensitive to the choice of split measure and even random splits do not significantly decrease accuracy. Buntine and Niblett (1992) present additional results on further domains and conclude that while random splitting leads to inferior trees, the information gain and Gini index measures are statistically indistinguishable.

The top-down methodology has the advantage of evaluating a potential attribute for a split in the context of the attributes associated with the nodes above it. The local greedy measures, however, consider each of the remaining attributes independently, ignoring potential interaction between different attributes (Mingers, 1989; Kononenko et al., 1997; Kim and Loh, 2001). We refer to the family of learning tasks where the utility of a set of attributes cannot be recognized by examining only subsets of it as tasks with a *strong interdependency*. When learning a problem with a strong interdependency, greedy measures can lead to a choice of non-optimal splits. To illustrate the above, let us consider the 2-XOR problem $a_1 \oplus a_2$ with two additional irrelevant attributes, a_3 and a_4. Assume that the set of examples is as listed in Figure 1(a). We observe that gain-1 of the irrelevant attribute a_4 is the highest:

$$0.13 = \text{gain}_1(a_4) > \text{gain}_1(a_1) = \text{gain}_1(a_2) = \text{gain}_1(a_3) = 0.02,$$

and hence ID3 would choose attribute a4 first. Figure 1(b) gives the decision tree as produced by ID3. Any positive instance with value 0 for a_4 would be misclassified by this decision tree. In the general case of parity concepts, the information gain measure is unable to distinguish between the relevant and irrelevant attributes because neither has a positive gain. Consequently, the learner will grow an overcomplicated tree with splits on irrelevant variables that come either in addition to or instead of the desired splits.

The problem of finding the smallest consistent tree[1] is known to be NP-complete (Hyafil and Rivest, 1976; Murphy and McCraw, 1991). In many applications that deal with hard problems, we are ready to allocate many more resources than required by simple greedy algorithms, but still cannot afford algorithms of exponential complexity. One commonly proposed approach for hard problems is anytime algorithms (Boddy and Dean, 1994), which can trade computation speed for quality. Quinlan (1993, chap. 11) recognized the need for this type of anytime algorithm for decision tree learning: "What is wanted is a resource constrained algorithm that will do the best it can within a specified computational budget and can pick up threads and continue if this budget is increased. This would make a challenging thesis topic!"

There are two main classes of anytime algorithms, namely *contract* and *interruptible* (Russell and Zilberstein, 1996). A contract algorithm is one that gets its resource allocation as a parameter. If interrupted at any point before the allocation is completed, it might not yield any useful results. An interruptible algorithm is one whose resource allocation is not given in advance and thus must be prepared to be interrupted at any moment. While the assumption of preallocated resources holds for many induction tasks, in many other real-life applications it is not possible to allocate the resources in advance. Therefore, in our work, we are interested both in contract and interruptible decision tree learners.

In this research, we suggest exploiting additional time resources by performing lookahead. Lookahead search is a well-known technique for improving greedy algorithms (Sarkar et al., 1994). When applied to decision tree induction, lookahead attempts to predict the profitability of a split at a node by estimating its effect on deeper descendants of the node. One of the main disadvantages of the greedy top-down strategy is that the effect of a wrong split decision is propagated down to all the nodes below it (Hastie et al., 2001, chap. 9). Lookahead search attempts to predict and avoid such non-contributive splits during the process of induction, before the final decision at each node is made.

Lookahead techniques have been applied to decision tree induction by several researchers. The reported results vary from *lookahead produces better trees* (Norton, 1989; Ragavan and Rendell, 1993; Dong and Kothari, 2001) to *lookahead does not help and can hurt* (Murthy and Salzberg, 1995). One problem with these works is their use of a uniform, fixed low-depth lookahead, therefore disqualifying the proposed algorithms from serving as anytime algorithms. Another problem is the data sets on which the lookahead methods were evaluated. For simple learning tasks, such as induction of conjunctive concepts, greedy methods perform quite well and no lookahead is needed. However, for more difficult concepts such as XOR, the greedy approach is likely to fail. Few other

1. A consistent decision tree is a tree that correctly classifies all training examples.

```
Procedure TDIDT($E, A$)
    If $E = \emptyset$ Return Leaf($nil$)
    If $\exists c$ such that $\forall e \in E$ Class($e$) $= c$
        Return Leaf($c$)
    $a \leftarrow$ CHOOSE-ATTRIBUTE($A, E$)
    $V \leftarrow$ domain($a$)
    Foreach $v_i \in V$
        $E_i \leftarrow \{e \in E \mid a(e) = v_i\}$
        $S_i \leftarrow$ TDIDT($E_i, A - \{a\}$)
    Return Node($a, \{\langle v_i, S_i \rangle \mid i = 1 \ldots |V|\}$)
```

Figure 2: Procedure for top-down induction of decision trees. E stands for the set of examples and A stands for the set of attributes.

non-greedy decision tree learners have been recently introduced (Bennett, 1994; Utgoff et al., 1997; Papagelis and Kalles, 2001; Page and Ray, 2003). These works, however, are not capable to handle high-dimensional difficult concepts and are not designed to offer anytime behavior.[2] The main challenge we face in this work is to make use of extra resources to induce better trees for hard concepts. Note that in contrast to incremental induction (Utgoff, 1989), we restrict ourselves in this paper to batch setup where all the training instances are available beforehand.

2. Contract Anytime Induction of Decision Trees

TDIDT (top-down induction of decision trees) methods start from the entire set of training examples, partition it into subsets by testing the value of an attribute, and then recursively call the induction algorithm for each subset. Figure 2 formalizes the basic algorithm for TDIDT. We focus first on consistent trees for which the stopping criterion for the top-down recursion is when all the examples have the same class label. Later, we consider pruning, which allows simpler trees at the cost of possible inconsistency with the training data (Breiman et al., 1984; Quinlan, 1993).

In this work we propose investing more time resources for making better split decisions. We first discuss tree size as a desired property of the tree to be learned and then we describe an anytime algorithm that uses sampling methods to obtain smaller trees.

2.1 Inductive Bias in Decision Tree Induction

The hypothesis space of TDIDT is huge and a major question is what strategy should be followed to direct the search. In other words, we need to decide what our *preference bias* (Mitchell, 1997, chap. 3) will be. This preference bias will be expressed in the CHOOSE-ATTRIBUTE procedure that determines which tree is explored next.

Ultimately, we would like to follow a policy that maximizes the accuracy of the tree on unseen examples. However, since these examples are not available, a heuristic should be used. Motivated by Occam's Razor, a widely adopted approach is to prefer smaller trees. The utility of this principle

2. In Section 5 we discuss related works in details.

to machine learning algorithms has been the subject of a heated debate. Several studies attempted to justify Occam's razor with theoretical and empirical arguments (Blumer et al., 1987; Quinlan and Rivest, 1989; Fayyad and Irani, 1990). But a number of recent works have questioned the utility of Occam's razor, and provided theoretical and experimental evidence against it.

Quinlan and Cameron-Jones (1995) provided empirical evidence that oversearching might result in less accurate rules. Experimental results with several UCI data sets indicate that the complexity of the produced theories does not correlate well with their accuracy, a finding that is inconsistent with Occam's Razor. Schaffer (1994) proved that no learning bias can outperform another bias over the space of all possible learning tasks. This looks like theoretical evidence against Occam's razor. Rao et al. (1995), however, argued against the applicability of this result to real-world problems by questioning the validity of its basic assumption about the uniform distribution of possible learning tasks. Webb (1996) presented C4.5X, an extension to C4.5 that uses similarity considerations to further specialize consistent leaves. Webb reported an empirical evaluation which shows that C4.5X has a slight advantage in a few domains and argued that these results discredit Occam's thesis.

Murphy and Pazzani (1994) reported a set of experiments in which all the possible consistent decision trees were produced and showed that, for several tasks, the smallest consistent decision tree had higher predictive error than slightly larger trees. However, when the authors compared the likelihood of better generalization for smaller vs. more complex trees, they concluded that simpler hypotheses should be preferred when no further knowledge of the target concept is available. The small number of training examples relative to the size of the tree that perfectly describes the concept might explain why, in these cases, the smallest tree did not generalize best. Another reason could be that only small spaces of decision trees were explored. To verify these explanations, we use a similar experimental setup where the data sets have larger training sets and attribute vectors of higher dimensionality. Because the number of all possible consistent trees is huge, we use a *Random Tree Generator* (RTG) to sample the space of trees obtainable by TDIDT algorithms. RTG builds a tree top-down and chooses the splitting attribute at random.

We report the results for three data sets: *XOR-5*, *Tic-tac-toe*, and *Zoo* (See Appendix A for detailed descriptions of these data sets). For each data set, the examples were partitioned into a training set (90%) and a testing set (10%), and RTG was used to generate a sample of 10 million consistent decision trees. The behavior in the three data sets is similar: the accuracy monotonically decreases with the increase in the size of the trees (number of leaves), confirming the utility of Occam's Razor. Further experiments with a variety of data sets indicate that the inverse correlation between size and accuracy is statistically significant (Esmeir and Markovitch, 2007).

It is important to note that smaller trees have several advantages aside from their probable greater accuracy, such as greater statistical evidence at the leaves, better comprehensibility, lower storage costs, and faster classification (in terms of total attribute evaluations).

Motivated by the above discussion, our goal is to find the smallest consistent tree. In TDIDT, the learner has to choose a split at each node, given a set of examples E that reach the node and a set of unused attributes A. For each attribute a, let $T_{min}(a,A,E)$ be the smallest tree rooted at a that is consistent with E and uses attributes from $A-\{a\}$ for the internal splits. Given two candidate attributes a_1 and a_2, we obviously prefer the attribute whose associated $T_{min}(a)$ is the smaller. For a set of attributes A, we define $\widetilde{A}$ to be the set of attributes whose associated $T_{min}(a)$ is the smaller. Formally, $\widetilde{A} = \arg\min_{a \in A} T_{min}(a)$. We say that a splitting attribute a is *optimal* if $a \in \widetilde{A}$. Observe that if the learner makes an optimal decision at each node, then the final tree is necessarily globally optimal.

```
Procedure ENTROPY-K($E, A, a, k$)
    If $k = 0$
        Return $\mathrm{I}(\mathrm{P}_E(c_1), \ldots, \mathrm{P}_E(c_n))$
    $V \leftarrow \mathrm{domain}(a)$
    Foreach $v_i \in V$
        $E_i \leftarrow \{e \in E \mid a(e) = v_i\}$
        Foreach $a' \in A$
            $A' \leftarrow A - \{a'\}$
            $h_i(a') \leftarrow$ ENTROPY-K($E_i, A', a', k-1$))
    Return $\sum_{i=1}^{|V|} \frac{|E_i|}{|E|} \min_{a' \in A} (h_i(a'))$

Procedure GAIN-K($E, A, a, k$)
    Return $\mathrm{I}(\mathrm{P}_E(c_1), \ldots, \mathrm{P}_E(c_n)) -$
                ENTROPY-K($E, A, a, k$)
```

Figure 3: Procedures for computing entropy$_k$ and gain$k$ for attribute a

2.2 Fixed-depth Lookahead

One possible approach for improving greedy TDIDT algorithms is to lookahead in order to examine the effect of a split deeper in the tree. ID3 uses entropy to test the effect of using an attribute one level below the current node. This can be extended to allow measuring the entropy at any depth k below the current node. This approach was the basis for the IDX algorithm (Norton, 1989). The recursive definition minimizes the $k-1$ entropy for each child and computes their weighted average. Figure 3 describes an algorithm for computing entropy$_k$ and its associated gain$_k$. Note that the gain computed by ID3 is equivalent to gain$_k$ for $k = 1$. We refer to this lookahead-based variation of ID3 as *ID3-k*. At each tree node, ID3-k chooses the attribute that maximizes gain$_k$.[3]

ID3-k can be viewed as a contract anytime algorithm parameterized by k. However, despite its ability to exploit additional resources when available, the anytime behavior of ID3-k is problematic. The run time of ID3-k grows exponentially as k increases.[4] As a result, the gap between the points of time at which the resulting tree can improve grows wider, limiting the algorithm's flexibility. Furthermore, it is quite possible that looking ahead to depth k will not be sufficient to find an optimal split. Entropy$_k$ measures the weighted average of the entropy in depth k but does not give a direct estimation of the resulting tree size. Thus, when $k < |A|$, this heuristic is more informed than entropy$_1$ but can still produce misleading results. In most cases we do not know in advance what value of k would be sufficient for correctly learning the target concept. Invoking exhaustive lookahead, that is, lookahead to depth $k = |A|$, will obviously lead to optimal splits, but its computational costs are prohibitively high. In the following subsection, we propose an alternative approach for evaluating attributes that overcomes the above-mentioned drawbacks of ID3-k.

3. If two attributes yield the same decrease in entropy, we prefer the one whose associated lookahead tree is shallower.
4. For example, in the binary case, ID3-k explores $\prod_{i=0}^{k-1} (n-i)^{2^i}$ combinations of attributes.

```
Procedure SID3-CHOOSE-ATTRIBUTE($E,A$)
    Foreach $a \in A$
        $p(a) \leftarrow \text{gain}_1(E,a)$
    If $\exists a$ such that $\text{entropy}_1(E,a) = 0$
        $a^* \leftarrow$ Choose attribute at random from
                $\{a \in A \mid \text{entropy}_1(E,a) = 0\}$
    Else
        $a^* \leftarrow$ Choose attribute at random from $A$;
                for each attribute $a$, the probability
                of selecting it is proportional to $p(a)$
    Return $a^*$
```

Figure 4: Attribute selection in SID3

2.3 Estimating Tree Size by Sampling

Motivated by the advantages of smaller decision trees, we introduce a novel algorithm that, given an attribute a, evaluates it by estimating the size of the minimal tree under it. The estimation is based on Monte-Carlo sampling of the space of consistent trees rooted at a. We estimate the minimum by the size of the smallest tree in the sample. The number of trees in the sample depends on the available resources, where the quality of the estimation is expected to improve with the increased sample size.

One way to sample the space of trees is to repeatedly produce random consistent trees using the RTG procedure. Since the space of consistent decision trees is large, such a sample might be a poor representative and the resulting estimation inaccurate. We propose an alternative sampling approach that produces trees of smaller expected size. Such a sample is likely to lead to a better estimation of the minimum.

Our approach is based on a new tree generation algorithm that we designed, called *Stochastic ID3* (SID3). Using this algorithm repeatedly allows the space of "good" trees to be sampled semi-randomly. In SID3, rather than choose an attribute that maximizes the information gain, we choose the splitting attribute randomly. The likelihood that an attribute will be chosen is proportional to its information gain.[5] However, if there are attributes that decrease the entropy to zero, then one of them is picked randomly. The attribute selection procedure of SID3 is listed in Figure 4.

To show that SID3 is a better sampler than RTG, we repeated our sampling experiment (Section 2.1) using SID3 as the sample generator. Figure 5 compares the frequency curves of RTG and SID3. Relative to RTG, the graph for SID3 is shifted to the left, indicating that the SID3 trees are clearly smaller. Next, we compared the average minimum found for samples of different sizes. Figure 6 shows the results. For the three data sets, the minimal size found by SID3 is strictly smaller than the value found by RTG. Given the same budget of time, RTG produced, on average, samples that are twice as large as that of SID3. However, even when the results are normalized (dashed line), SID3 is still superior.

Having decided about the sampler, we are ready to describe our proposed contract algorithm, *Lookahead-by-Stochastic-ID3* (LSID3). In LSID3, each candidate split is evaluated by the estimated

5. We make sure that attributes with gain of zero will have a positive probability of being selected.

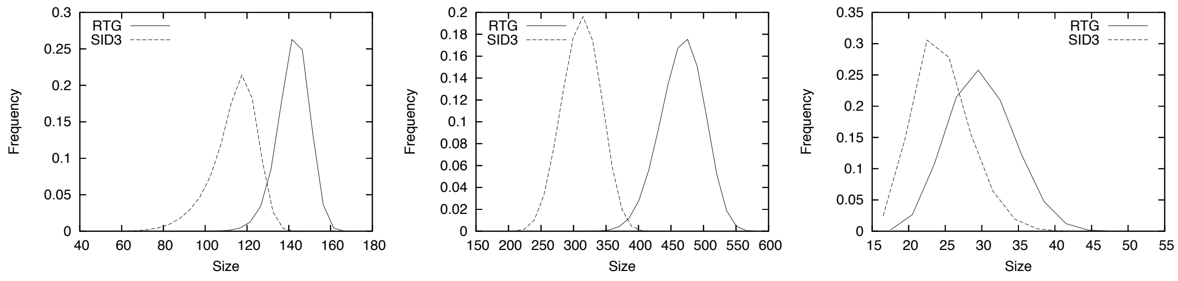


Figure 5: Frequency curves for the *XOR-5* (left), *Tic-tac-toe*, and *Zoo* (right) data sets

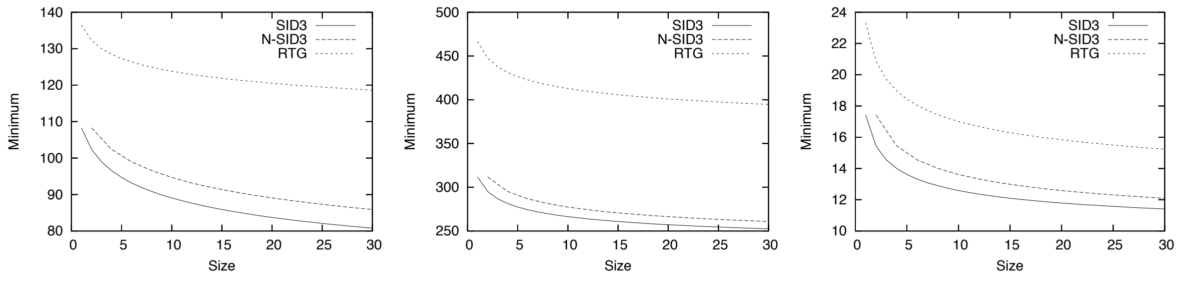


Figure 6: The minimum as estimated by SID3 and RTG as a function of the sample size. The data sets are *XOR5* (left), *Tic-tac-toe* and *Zoo* (right). The dashed line describes the results for SID3 normalized by time.

size of the subtree under it. To estimate the size under an attribute a, LSID3 partitions the set of examples according to the values a can take and repeatedly invokes SID3 to sample the space of trees consistent with each subset. Summing up the minimal tree size for each subset gives an estimation of the minimal total tree size under a.

LSID3 is a contract algorithm parameterized by r, the sample size. LSID3 with $r = 0$ is defined to choose the splitting attribute using the standard ID3 selection method. Figure 7 illustrates the choice of splitting attributes as made by LSID3. In the given example, SID3 is called twice for each subset and the evaluation of the examined attribute a is the sum of the two minima: $min(4,3) + min(2,6) = 5$. The method for choosing a splitting attribute is formalized in Figure 8.

To analyze the time complexity of LSID3, let m be the total number of examples and n be the total number of attributes. For a given node y, we denote by n_y the number of candidate attributes at y, and by m_y the number of examples that reach y. ID3, at each node y, calculates gain for n_y attributes using m_y examples, that is, the complexity of choosing an attribute is $O(n_y \cdot m_y)$. At level i of the tree, the total number of examples is bounded by m and the number of attributes to consider is $n - i$. Thus, it takes $O(m \cdot (n - i))$ to find the splits for all nodes in level i. In the worst case the tree will be of depth n and hence the total runtime complexity of ID3 will be $O(m \cdot n^2)$ (Utgoff, 1989). Shavlik et al. (1991) reported for ID3 an empirically based average-case complexity of $O(m \cdot n)$.

It is easy to see that the complexity of SID3 is similar to that of ID3. LSID3(r) invokes SID3 r times for each candidate split. Recalling the above analysis for the time complexity of ID3, we can

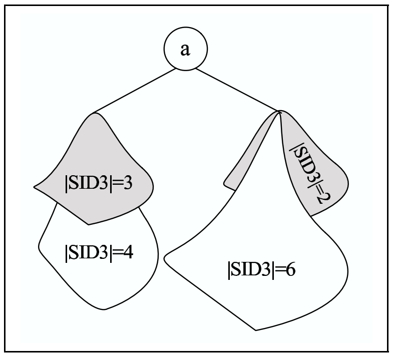


Figure 7: Attribute evaluation using LSID3. The estimated subtree size for a is $min(4,3) + min(2,6) = 5$.

```
Procedure LSID3-CHOOSE-ATTRIBUTE($E, A, r$)
    If $r = 0$
        Return ID3-CHOOSE-ATTRIBUTE(E, A)
    Foreach $a \in A$
        Foreach $v_i \in$ domain($a$)
            $E_i \leftarrow \{e \in E \mid a(e) = v_i\}$
            $min_i \leftarrow \infty$
            Repeat $r$ times
                $min_i \leftarrow \min(min_i, |\text{SID3}(E_i, A - \{a\})|)$
        $total_a \leftarrow \sum_{i=1}^{|\text{domain}(a)|} min_i$
    Return $a$ for which $total_a$ is minimal
```

Figure 8: Attribute selection in LSID3

write the complexity of LSID3(r) as

$$\sum_{i=0}^{n-1} r \cdot (n-i) \cdot O(m \cdot (n-i)^2) = \sum_{i=1}^{n} O(r \cdot m \cdot i^3) = O(r \cdot m \cdot n^4).$$

In the average case, we replace the runtime of SID3 by $O(m \cdot (n-i))$, and hence we have

$$\sum_{i=0}^{n-1} r \cdot (n-i) \cdot O(m \cdot (n-i)) = \sum_{i=1}^{n} O(r \cdot m \cdot i^2) = O(r \cdot m \cdot n^3). \qquad (1)$$

According to the above analysis, the run-time of LSID3 grows at most linearly with r (under the assumption that increasing r does not result in larger trees). We expect that increasing r will improve the classifier's quality. To understand why, let us examine the expected behavior of the LSID3 algorithm on the 2-XOR problem used in Figure 1(b). LSID3 with $r = 0$, which is equivalent to ID3, prefers to split on the irrelevant attribute a_4. LSID3 with $r \geq 1$ evaluates each attribute a by calling SID3 to estimate the size of the trees rooted at a. The attribute with the smallest estimation will be selected. The minimal size for trees rooted at a_4 is 6 and for trees rooted at a_3 is 7. For a_1 and a_2, SID3 would necessarily produce trees of size 4.[6] Thus, LSID3, even with $r = 1$, will succeed in learning the right tree. For more complicated problems such as 3-XOR, the space of SID3 trees under the relevant attributes includes trees other than the smallest. In that case, the larger the sample is, the higher the likelihood is that the smallest tree will be drawn.

2.4 Evaluating Continuous Attributes by Sampling the Candidate Cuts

When attributes have a continuous domain, the decision tree learner needs to discretize their values in order to define splitting tests. One common approach is to partition the range of values an attribute can take into bins, prior to the process of induction. Dougherty et al. (1995) review and compare several different strategies for pre-discretization. Using such a strategy, our lookahead algorithm can operate unchanged. Pre-discretization, however, may be harmful in many domains because the correct partitioning may differ from one context (node) to another.

An alternative approach is to determine dynamically at each node a threshold that splits the examples into two subsets. The number of different possible thresholds is at most $|E| - 1$, and thus the number of candidate tests to consider for each continuous attribute increases to $O(|E|)$. Such an increase in complexity may be insignificant for greedy algorithms, where evaluating each split requires a cheap computation (like information gain in ID3). In LSID3, however, the dynamic method may present a significant problem because evaluating the usefulness of each candidate split is very costly.

The desired method would reduce the number of splits to evaluate while avoiding the disadvantages of pre-discretization. We devised a method for controlling the resources devoted to evaluating a continuous attribute by Monte Carlo sampling of the space of splits. Initially, we evaluate each possible splitting test by the information gain it yields. This can be done in linear time $O(|E|)$. Next, we choose a sample of the candidate splitting tests where the probability that a test will be chosen is proportional to its information gain. Each candidate in the sample is evaluated by a single invocation of SID3. However, since we sample with repetitions, candidates with high information gain may have multiple instances in the sample, resulting in several SID3 invocations.

The resources allocated for evaluating a continuous attribute using the above method are determined by the sample size. If our goal is to devote a similar amount of resources to all attributes, then we can use r as the size of the sample. Such an approach, however, does not take into account the size of the population to be sampled. We use a simple alternative approach of taking samples with size $p \cdot |E|$ where p is a predetermined parameter, set by the user according to the available resources.[7] Note that p can be greater than one because we sample with repetition.

We name this variant of the LSID3 algorithm LSID3-MC, and formalize it in Figure 9. LSID3-MC can serve as an anytime algorithm that is parameterized by p and r.

2.5 Multiway vs. Binary Splits

The TDIDT procedure, as described in Figure 2, partitions the current set of examples into subsets according to the different values the splitting attribute can take. LSID3, which is a TDIDT algorithm, inherits this property. While multiway splits can yield more comprehensible trees (Kim and

6. Neither a_3 nor a_4 can be selected at the 2^{nd} level since the remaining relevant attribute reduces the entropy to zero.
7. Similarly to the parameter r, a mapping from the available time to p is needed (see Section 2.7).

```
Procedure MC-EVALUATE-CONTINUOUS-ATTRIBUTE($(E, A, a, p)$)
    Foreach $e \in E$
        $E_{\leq} \leftarrow \{e' \mid a(e') \leq a(e)\}$
        $E_{>} \leftarrow \{e' \mid a(e') > a(e)\}$
        $entropy_e \leftarrow \frac{|E_{\leq}|}{|E|}\text{entropy}(E_{\leq}) + \frac{|E_{>}|}{|E|}\text{entropy}(E_{>})$
        $gain_e \leftarrow \text{entropy}(E) - entropy_e$
    $sampleSize \leftarrow p \cdot |E|$
    $min \leftarrow \infty$
    Repeat $sampleSize$ times
        $e \leftarrow$ Choose an example at random from $E$; for each example $e$,
              the probability of selecting it is proportional to $gain_e$
        $E_{\leq} \leftarrow \{e' \mid a(e') \leq a(e)\}$
        $E_{>} \leftarrow \{e' \mid a(e') > a(e)\}$
        $total_e \leftarrow |\text{SID3}(E_{\leq}, A)| + |\text{SID3}(E_{>}, A)|$
        If $total_e < min$
            $min \leftarrow total_e$
            $bestTest \leftarrow a(e)$
    Return $\langle min, bestTest \rangle$
```

Figure 9: Monte Carlo evaluation of continuous attributes using SID3

Loh, 2001), they might fragment the data too quickly, decreasing its learnability (Hastie et al., 2001, chap. 9).

To avoid the fragmentation problem, several learners take a different approach and consider only binary splits (e.g., CART and QUEST, Loh and Shih, 1997). One way to obtain binary splits when an attribute a is categorical is to partition the examples using a single value v: all the examples with $a = v$ are bagged into one group and all the other examples are bagged into a second group. Therefore, for each attribute a there are $|\text{domain}(a)|$ candidate splits. While efficient, this strategy does not allow different values to be combined, and therefore might over-fragment the data. CART overcomes this by searching over all non-empty subsets of $\text{domain}(a)$. We denote by BLSID3 a variant of LSID3 that yields binary trees and adopts the CART style exhaustive search. Observe that the complexity of evaluating an attribute in BLSID3 is $O(2^{|\text{domain}(a)|})$ times that in LSID3.

A second problem with LSID3's multiway splits is their bias in attribute selection. Consider for example a learning task with four attributes: three binary attributes a_1, a_2, a_3, and a categorical attribute a_4 whose domain is $\{A, C, G, T\}$. Assume that the concept is $a_1 \oplus a_2$ and that a_4 was created from the values of a_2 by randomly and uniformly mapping every 0 into A or C and every 1 into G or T. Theoretically, a_2 and a_4 are equally important to the class. LSID3, however, would prefer to split a_2 due to its smaller associated total tree size.[8] BLSID3 solves this problem by considering the binary test $\in \{A, C\}$.

Loh and Shih (1997) pointed out that exhaustive search tends to prefer features with larger domains and proposed QUEST, which overcomes this problem by dealing with attribute selection and with test selection separately: first an attribute is chosen and only then is the best test picked.

8. Note that this bias is opposite to that of favoring attributes with a large domain (Quinlan, 1993, chap. 2).

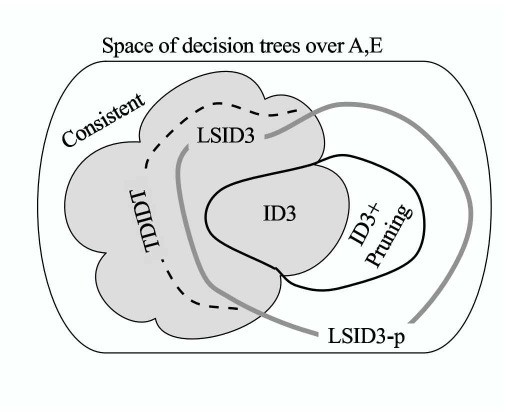


Figure 10: The space of decision trees over a set of attributes *A* and a set of examples *E*

This bias, and its reduction by QUEST, were studied in the context of greedy attribute evaluation, but can potentially be applied to our non-greedy approach.

2.6 Pruning the LSID3 Trees

Pruning tackles the problem of how to avoid overfitting the data, mainly in the presence of classification noise. It is not designed, however, to recover from wrong greedy decisions. When there are sufficient examples, wrong splits do not significantly diminish accuracy, but result in less comprehensible trees and costlier classification. When there are fewer examples the problem is intensified because the accuracy is adversely affected. In our proposed anytime approach, this problem is tackled by allotting additional time for reaching trees unexplored by greedy algorithms with pruning. Therefore, we view pruning as orthogonal to lookahead and suggest considering it as an additional phase in order to avoid overfitting.

Pruning can be applied in two different ways. First, it can be applied as a second phase after each LSID3 final tree has been built. We denote this extension by *LSID3-p* (and *BLSID3-p* for binary splits). Second, it is worthwhile to examine pruning of the lookahead trees. For this purpose, a post-pruning phase will be applied to the trees that were generated by SID3 to form the lookahead sample, and the size of the pruned trees will be considered. The expected contribution of pruning in this case is questionable since it has a similar effect on the entire lookahead sample.

Previous comparative studies did not find a single pruning method that is generally the best and conclude that different pruning techniques behave similarly (Esposito et al., 1997; Oates and Jensen, 1997). Therefore, in our experiments we adopt the same pruning method as in C4.5, namely error-based pruning (EPB), and examine post-pruning of the final trees as well as pruning of the lookahead trees.

Figure 10 describes the spaces of decision trees obtainable by LSID3. Let *A* be a set of attributes and *E* be a set of examples. The space of decision trees over *A* can be partitioned into two: the space of trees consistent with *E* and the space of trees inconsistent with *E*. The space of consistent trees includes as a subspace the trees inducible by TDIDT.[9] The trees that ID3 can produce are a subspace of the TDIDT space. Pruning extends the space of ID3 trees to the inconsistent space. LSID3

9. TDIDT is a strict subspace because it cannot induce trees that: (1) contain a subtree with all leaves marked with the same class, and (2) include an internal node that has no associated training examples.

can reach trees that are obtainable by TDIDT but not by the greedy methods and their extensions. LSID3-p expands the search space to trees that do not fit the training data perfectly, so that noisy training sets can be handled.

2.7 Mapping Contract Time to Sample Size

LSID3 is parameterized by r, the number of times SID3 is invoked for each candidate. In real-life applications, however, the user supplies the system with the time she is willing to allocate for the learning process. Hence, a mapping from the contract time to r is needed.

One way to obtain such a mapping is to run the algorithm with $r = 1$. Since the runtime grows linearly with r, one can now, given the remaining time and the runtime for $r = 1$, easily find the corresponding value of r. Then, the whole induction process is restarted with the measured r as the contract parameter. The problem with this method is that, in many cases, running the algorithm with $r = 1$ consumes many of the available resources while not contributing to the final classifier.

Another possibility is mapping time to r on the basis of previous experience on the same domain or a domain with similar properties. When no such knowledge exists, one can derive the runtime from the time complexity analysis. However, in practice, the runtime is highly dependent on the complexity of the domain and the number of splits that in fact take place. Hence it is very difficult to predict.

3. Interruptible Anytime Learning of Decision Trees

As a contract algorithm, LSID3 assumes that the contract parameter is known in advance. In many real-life cases, however, either the contract time is unknown or mapping the available resources to the appropriate contract parameter is not possible. To handle such cases, we need to devise an interruptible anytime algorithm.

We start with a method that converts LSID3 to an interruptible algorithm using the general sequencing technique described by Russell and Zilberstein (1996). This conversion, however, uses the contract algorithm as a black box and hence cannot take into account specific aspects of decision tree induction. Next, we present a novel iterative improvement algorithm, *Interruptible Induction of Decision Trees* (IIDT), that repeatedly replaces subtrees of the current hypothesis with subtrees generated with higher resource allocation and thus expected to be better.

3.1 Interruptible Induction by Sequencing Contract Algorithms

By definition, every interruptible algorithm can serve as a contract algorithm because one can stop the interruptible algorithm when all the resources have been consumed. Russell and Zilberstein (1996) showed that the other direction works as well: any contract algorithm $\mathcal{A}$ can be converted into an interruptible algorithm $\mathcal{B}$ with a constant penalty. $\mathcal{B}$ is constructed by running $\mathcal{A}$ repeatedly with exponentially increasing time limits. This general approach can be used to convert LSID3 into an interruptible algorithm. LSID3 gets its contract time in terms of r, the sample size. When $r = 0$, LSID3 is defined to be identical to ID3. Therefore, we first call LSID3 with $r = 0$ and then continue with exponentially increasing values of r, starting from $r = 1$. Figure 11 formalizes the resulting algorithm. It can be shown that the above sequence of runtimes is optimal when the different runs are scheduled on a single processor (Russell and Zilberstein, 1996).

```
Procedure SEQUENCED-LSID3($E,A$)
    $T \leftarrow$ ID3($E,A$)
    $r \leftarrow 1$
    While not-interrupted
        $T \leftarrow$ LSID3($E,A,r$)
        $r \leftarrow 2 \cdot r$
    Return $T$
```

Figure 11: Conversion of LSID3 to an interruptible algorithm by sequenced invocations

One problem with the sequencing approach is the exponential growth of the gaps between the times at which an improved result can be obtained. The reason for this problem is the generality of the sequencing approach, which views the contract algorithm as a black box. Thus, in the case of LSID3, at each iteration the whole decision tree is rebuilt. In addition, the minimal value that can be used for τ is the runtime of LSID3($r = 1$). In many cases we would like to stop the algorithm earlier. When we do, the sequencing approach will return the initial greedy tree. Our interruptible anytime framework, called IIDT, overcomes these problems by iteratively improving the tree rather than trying to rebuild it.

3.2 Interruptible Induction by Iterative Improvement

Iterative improvement approaches that start with an initial solution and repeatedly modify it have been successfully applied to many AI domains, such as CSP, timetabling, and combinatorial problems (Minton et al., 1992; Johnson et al., 1991; Schaerf, 1996). The key idea behind iterative improvement techniques is to choose an element of the current suboptimal solution and improve it by local repairs. IIDT adopts the above approach for decision tree learning and iteratively revises subtrees. It can serve as an interruptible anytime algorithm because, when interrupted, it can immediately return the currently best solution available.

As in LSID3, IIDT exploits additional resources in an attempt to produce better decision trees. The principal difference between the algorithms is that LSID3 uses the available resources to induce a decision tree top-down, where each decision made at a node is final and does not change. IIDT, however, is not allocated resources in advance and uses extra time resources to modify split decisions repeatedly.

IIDT receives a set of examples and a set of attributes. It first performs a quick construction of an initial decision tree by calling ID3. Then it iteratively attempts to improve the current tree by choosing a node and rebuilding its subtree with more resources than those used previously. If the newly induced subtree is better, it will replace the existing one. IIDT is formalized in Figure 12.

Figure 13 illustrates how IIDT works. The target concept is $a_1 \oplus a_2$, with two additional irrelevant attributes, a_3 and a_4. The leftmost tree was constructed using ID3. In the first iteration, the subtree rooted at the bolded node is selected for improvement and replaced by a smaller tree (surrounded by a dashed line). Next, the root is selected for improvement and the whole tree is replaced by a tree that perfectly describes the concept. While in this particular example IIDT first chooses to rebuild a subtree at depth 2 and then at depth 1, it considers all subtrees, regardless of their level.

```
Procedure IIDT($E,A$)
    $T \leftarrow$ ID3($E,A$)
    While not-interrupted
        $node \leftarrow$ CHOOSE-NODE($T,E,A$)
        $t \leftarrow$ subtree of $T$ rooted at $node$
        $A_{node} \leftarrow \{a \in A \mid a \notin$ ancestor of $node\}$
        $E_{node} \leftarrow \{e \in E \mid e$ reaches $node\}$
        $r \leftarrow$ NEXT-R($node$)
        $t' \leftarrow$ REBUILD-TREE($E_{node},A_{node},r$)
        If EVALUATE($t$) $>$ EVALUATE($t'$)
            replace $t$ with $t'$
    Return $T$
```

Figure 12: Interruptible induction of decision trees

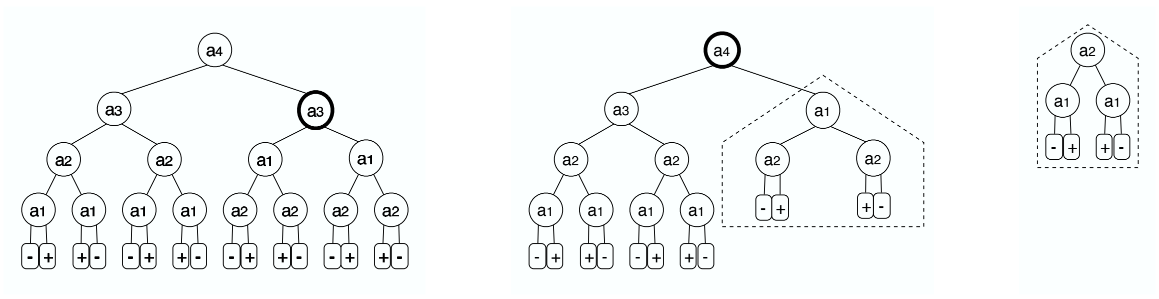


Figure 13: Iterative improvement of the decision tree produced for the 2-XOR concept $a_1 \oplus a_2$ with two additional irrelevant attributes, a_3 and a_4. The leftmost tree was constructed using ID3. In the first iteration the subtree rooted at the bolded node is selected for improvement and replaced by a smaller tree (surrounded by a dashed line). Next, the root is selected for improvement and the whole tree is replaced by a tree that perfectly describes the concept.

IIDT is designed as a general framework for interruptible learning of decision trees. It can use different approaches for choosing which node to improve, for allocating resources for an improvement iteration, for rebuilding a subtree, and for deciding whether an alternative subtree is better.

After the subtree to be rebuilt is chosen and the resources for a reconstruction iteration allocated, the problem becomes a task for a contract algorithm. A good candidate for such an algorithm is LSID3, which is expected to produce better subtrees when invoked with a higher resource allocation. In what follows we focus on the different components of IIDT and suggest a possible implementation that uses LSID3 for revising subtrees.

3.2.1 CHOOSING A SUBTREE TO IMPROVE

Intuitively, the next node we would like to improve is the one with the highest expected marginal utility, that is, the one with the highest ratio between the expected benefit and the expected cost (Hovitz, 1990; Russell and Wefald, 1989). Estimating the expected gain and expected cost of rebuilding a subtree is a difficult problem. There is no apparent way to estimate the expected improvement in terms of either tree size or generalization accuracy. In addition, the resources to be consumed by LSID3 are difficult to predict precisely. We now show how to approximate these values, and how to incorporate these approximations into the node selection procedure.

Resource Allocation. The LSID3 algorithm receives its resource allocation in terms of r, the number of samplings devoted to each attribute. Given a tree node y, we can view the task of rebuilding the subtree below y as an independent task. Every time y is selected, we have to allocate resources for the reconstruction process. Following Russell and Zilberstein (1996), the optimal strategy in this case is to double the amount of resources at each iteration. Thus, if the resources allocated for the last attempted improvement of y were $\text{LAST-R}(y)$, the next allocation will be $\text{NEXT-R}(y) = 2 \cdot \text{LAST-R}(y)$.

Expected Cost. The expected cost can be approximated using the average time complexity of the contract algorithm used to rebuild subtrees. Following Equation 1, we estimate $\text{NEXT-R}(y) \cdot m \cdot n^3$ to be the expected runtime of LSID3 when rebuilding a node y, where m is the number of examples that reach y and n is the number of attributes to consider. We observe that subtrees rooted in deeper levels are preferred because they have fewer examples and attributes to consider. Thus, their expected runtime is shorter. Furthermore, because each time allocation for a node doubles the previous one, nodes that have already been selected many times for improvement will have higher associated costs and are less likely to be chosen again.

Expected benefit. The whole framework of decision tree induction rests on the assumption that smaller consistent trees are better than larger ones. Therefore, the size of a subtree can serve as a measure for its quality. It is difficult, however, to estimate the size of the reconstructed subtree without actually building it. Therefore, we use instead an upper limit on the possible reduction in size. The minimal size possible for a decision tree is obtained when all examples are labelled with the same class. Such cases are easily recognized by the greedy ID3. Similarly, if a subtree were replaceable by another subtree of depth 1, ID3 (and LSID3) would have chosen the smaller subtree. Thus, the maximal reduction of the size of an existing subtree is to the size of a tree of depth 2. Assuming that the maximal number of values per attribute is b, the maximal size of such a tree is b^2. Hence, an upper bound on the benefit from reconstructing an existing tree t is $\text{SIZE}(t) - b^2$. Ignoring the expected costs and considering only the expected benefits results in giving the highest score to the root node. This makes sense: assuming that we have infinite resources, we would attempt to improve the entire tree rather than parts of it.

Granularity. Considering the cost and benefit approximations described above, the selection procedure would prefer deep nodes (that are expected to have low costs) with large subtrees (that are expected to yield large benefits). When no such large subtrees exist, our algorithm may repeatedly attempt to improve smaller trees rooted at deep nodes because these trees have low associated costs. In the short term, this behavior would indeed be beneficial but can be harmful in the long term. This is because when the algorithm later improves subtrees in upper levels, the resources spent on deeper

```
**Procedure** CHOOSE-NODE$(T, E, A, g)$
    $max\text{-}cost \leftarrow \text{NEXT-R}(root) \cdot |E| \cdot |A|^3$
    **Foreach** $node \in T$
        $A_{node} \leftarrow \{a \in A \mid a \notin \text{ ancestor of } node\}$
        $E_{node} \leftarrow \{e \in E \mid e \text{ reaches } node\}$
        $r_{node} \leftarrow \text{NEXT-R}(node)$
        $cost_{node} \leftarrow r_{node} \cdot |E_{node}| \cdot |A_{node}|^3$
        **If** $(cost_{node}/max\text{-}cost) > g$
            $l\text{-}bound \leftarrow (\min_{a \in A_{node}} |\text{DOMAIN}(a)|)^2$
            $\Delta q \leftarrow \text{LEAVES}(node) - l\text{-}bound$
            $u_{node} \leftarrow \Delta q / cost_{node}$
    $best \leftarrow node$ that maximizes $u_{node}$
    **Return** $\langle best, r_{best} \rangle$

**Procedure** NEXT-R$(node)$
    **If** $\text{LAST-R}(node) = 0$
        **Return** 1
    **Else**
        **Return** $2 \cdot \text{LAST-R}(node)$
```

Figure 14: Choosing a node for reconstruction

nodes will have been wasted. Had the algorithm first selected the upper level trees, this waste would have been avoided, but the time gaps between potential improvements would have increased.

To control the tradeoff between efficient resource use and anytime performance flexibility, we add a granularity parameter $0 \leq g \leq 1$. This parameter serves as a threshold for the minimal time allocation for an improvement phase. A node can be selected for improvement only if its normalized expected cost is above g. To compute the normalized expected cost, we divide the expected cost by the expected cost of the root node. Note that it is possible to have nodes with a cost that is higher than the cost of the root node, since the expected cost doubles the cost of the last improvement of the node. Therefore, the normalized expected cost can be higher than 1. Such nodes, however, will never be selected for improvement, because their expected benefit is necessarily lower than the expected benefit of the root node. Hence, when $g = 1$, IIDT is forced to choose the root node and its behavior becomes identical to that of the sequencing algorithm described in Section 3.1.

Figure 14 formalizes the procedure for choosing a node for reconstruction. Observe that IIDT does not determine g but expects the user to provide this value according to her needs: more frequent small improvements or faster overall progress.

3.2.2 EVALUATING A SUBTREE

Although LSID3 is expected to produce better trees when allocated more resources, an improved result is not guaranteed. Thus, to avoid obtaining an induced tree of lower quality, we replace an existing subtree with a newly induced alternative only if the alternative is expected to improve the quality of the complete decision tree. Following Occam's Razor, we measure the usefulness of a subtree by its size. Only if the reconstructed subtree is smaller does it replace an existing subtree. This guarantees that the size of the complete decision tree will decrease monotonically.

Another possible measure is the accuracy of the decision tree on a set-aside validation set of examples. In this case the training set is split into two subsets: a growing set and a validation set. Only if the accuracy on the validation set increases is the modification applied. This measure suffers from two drawbacks. The first is that putting aside a set of examples for validation results in a smaller set of training examples, making the learning process harder. The second is the bias towards overfitting the validation set, which might reduce the generalization abilities of the tree. Several of our experiments, which we do not report here, confirmed that relying on the tree size results in better decision trees.

4. Empirical Evaluation

A variety of experiments were conducted to test the performance and behavior of the proposed anytime algorithms. First we describe our experimental methodology and explain its motivation. We then present and discuss our results.

4.1 Experimental Methodology

We start our experimental evaluation by comparing our contract algorithm, given a fixed resource allocation, with the basic decision tree induction algorithms. We then compare the anytime behavior of our contract algorithm to that of fixed lookahead. Next we examine the anytime behavior of our interruptible algorithm. Finally, we compare its performance to several modern decision tree induction methods.

Following the recommendations of Bouckaert (2003), 10 runs of a 10-fold cross-validation experiment were conducted for each data set and the reported results averaged over the 100 individual runs.[10] For the *Monks* data sets, which were originally partitioned into a training set and a testing set, we report the results on the original partitions. Due to the stochastic nature of LSID3, the reported results in these cases are averaged over 10 different runs.

In order to evaluate the studied algorithms, we used 17 data sets taken from the UCI repository (Blake and Merz, 1998).[11] Because greedy learners perform quite well on easy tasks, we looked for problems that hide hard concepts so the advantage of our proposed methods will be emphasized. The UCI repository, nevertheless, contains only few such tasks. Therefore, we added 7 artificial ones.[12] Several commonly used UCI data sets were included (among the 17) to allow comparison with results reported in literature. Table 1 summarizes the characteristics of these data sets while Appendix A gives more detailed descriptions. To compare the performance of the different algorithms, we will consider two evaluation criteria over decision trees: (1) *generalization accuracy*, measured by the ratio of the correctly classified examples in the testing set, and (2) *tree size*, measured by the number of non-empty leaves in the tree.

10. An exception was the *Connect-4* data set, for which only one run of 10-fold CV was conducted because of its enormous size.
11. Due to the time-intensiveness of the experiments, we limited ourselves to 17 UCI problems.
12. The artificial data sets are available at http://www.cs.technion.ac.il/∼esaher/publications/datasets.

		ATTRIBUTES		MAX ATTRIBUTE	
DATA SET	INSTANCES	NOMINAL (BINARY)	NUMERIC	DOMAIN	CLASSES
AUTOMOBILE - MAKE	160	10 (4)	15	8	22
AUTOMOBILE - SYMBOLING	160	10 (4)	15	22	7
BALANCE SCALE	625	4 (0)	0	5	3
BREAST CANCER	277	9 (3)	0	13	2
CONNECT-4	68557	42 (0)	0	3	3
CORRAL	32	6 (6)	0	2	2
GLASS	214	0 (0)	9	-	7
IRIS	150	0 (0)	4	-	3
MONKS-1	124+432	6 (2)	0	4	2
MONKS-2	169+432	6 (2)	0	4	2
MONKS-3	122+432	6 (2)	0	4	2
MUSHROOM	8124	22 (4)	0	12	2
SOLAR FLARE	323	10 (5)	0	7	4
TIC-TAC-TOE	958	9 (0)	0	3	2
VOTING	232	16 (16)	0	2	2
WINE	178	0 (0)	13	-	3
ZOO	101	16 (15)	0	6	7
NUMERIC XOR 3D	200	0 (0)	6	-	2
NUMERIC XOR 4D	500	0 (0)	8	-	2
MULTIPLEXER-20	615	20 (20)	0	2	2
MULTIPLEX-XOR	200	11 (11)	0	2	2
XOR-5	200	10 (10)	0	2	2
XOR-5 10% NOISE	200	10 (10)	0	2	2
XOR-10	10000	20 (20)	0	2	2

Table 1: Characteristics of the data sets used

4.2 Fixed Time Comparison

Our first set of experiments compares ID3, C4.5 with its default parameters, ID3-k($k = 2$), LSID3($r = 5$) and LSID3-p($r = 5$). We used our own implementation for all algorithms, where the results of C4.5 were validated with WEKA's implementation (Witten and Frank, 2005). We first discuss the results for the consistent trees and continue by analyzing the findings when pruning is applied.

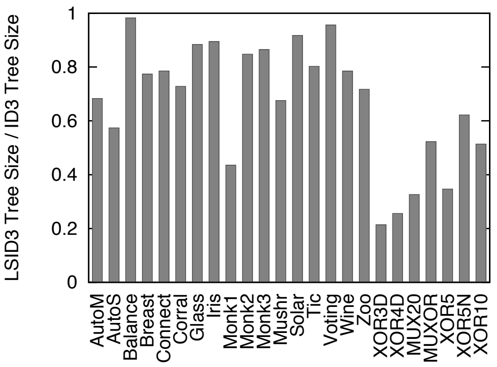
4.2.1 CONSISTENT TREES

Figure 15 illustrates the differences in tree size and generalization accuracy of LSID3(5) and ID3. Figure 16 compares the performance of LSID3 to that of ID3-k. The full results, including significance tests, are available in Appendix B.

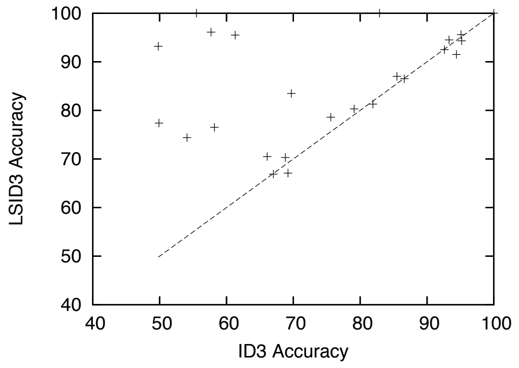
When comparing the algorithms that produce consistent trees, namely ID3, ID3-k and LSID3, the average tree size is the smallest for most data sets when the trees are induced with LSID3. In all cases, as Figure 15(a) implies, LSID3 produced smaller trees than ID3 and these improvements were found to be significant. The average reduction in size is 26.5% and for some data sets, such as *XOR-5* and *Multiplexer-20*, it is more than 50%. ID3-k produced smaller trees than ID3 for most but not all of the data sets (see Figure 16, a).

In the case of synthetic data sets, the optimal tree size can be found in theory.[13] For instance, the tree that perfectly describes the n XOR concept is of size 2^n. The results show that in this sense, the trees induced by LSID3 were almost optimal.

13. Note that a theoretically optimal tree is not necessarily obtainable from a given training set.

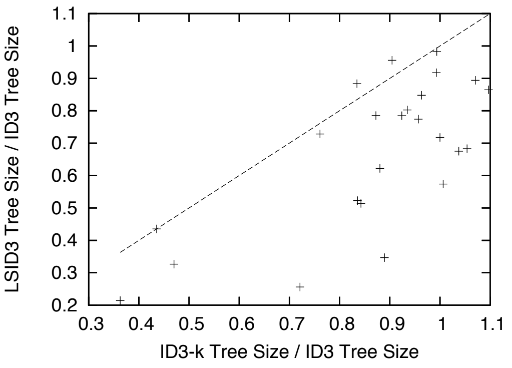


(a) Tree Size

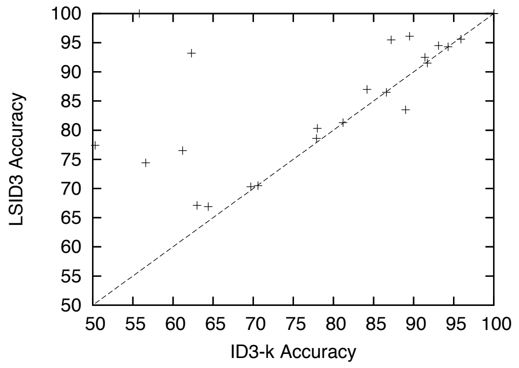


(b) Accuracy

Figure 15: Illustration of the differences in performance between LSID3(5) and ID3. The left-side figure gives the relative size of the trees produced by LSID3 in comparison to ID3. The right-side figure plots the accuracy achieved by both algorithms. Each point represents a data set. The *x*-axis represents the accuracy of ID3 while the *y*-axis represents that of LSID3. The dashed line indicates equality. Points are above it if LSID3 performs better and below it if ID3 is better.



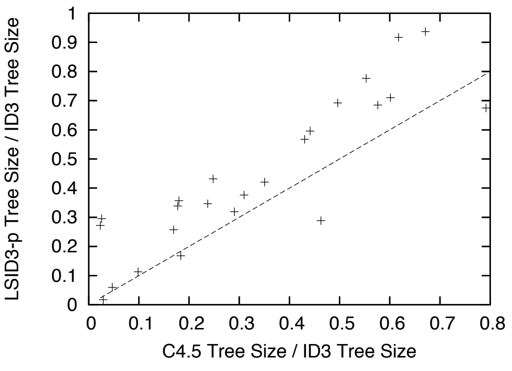
(a) Tree Size



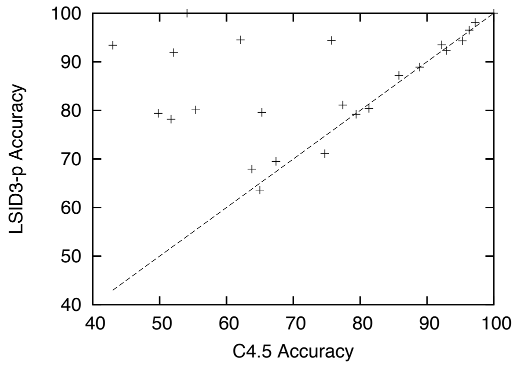
(b) Accuracy

Figure 16: Performance differences for LSID3(5) and ID3-k. The left-side figure compares the size of trees induced by each algorithm, measured relative to ID3 (in percents). The right-side figure plots the absolute differences in terms of accuracy.

Reducing the tree size is usually beneficial only if the associated accuracy is not reduced. Analyzing the accuracy of the produced trees shows that LSID3 significantly outperforms ID3 for most data sets. For the other data sets, the t-test values indicate that the algorithms are not significantly



(a) Tree Size



(b) Accuracy

Figure 17: Performance differences for LSID3-p(5) and C4.5. The left-side figure compares the size of trees induced by each algorithm, measured relative to ID3 (in percents). The right-side figure plots the absolute differences in terms of accuracy.

different. The average absolute improvement in the accuracy of LSID3 over ID3 is 11%. The Wilcoxon test (Demsar, 2006), which compares classifiers over multiple data sets, indicates that the advantage of LSID3 over ID3 is significant, with $\alpha = 0.05$.

The accuracy achieved by ID3-k, as shown in Figure 16(b), is better than that of ID3 on some data sets. ID3-k achieved similar results to LSID3 for some data sets, but performed much worse for others, such as *Tic-tac-toe* and *XOR-10*. For most data sets, the decrease in the size of the trees induced by LSID3 is accompanied by an increase in predictive power. This phenomenon is consistent with Occam's Razor.

4.2.2 PRUNED TREES

Pruning techniques help to avoid overfitting. We view pruning as orthogonal to our lookahead approach. Thus, to allow handling noisy data sets, we tested the performance of LSID3-p, which post-prunes the LSID3 trees using error-based pruning.

Figure 17 compares the performance of LSID3-p to that of C4.5. Applying pruning on the trees induced by LSID3 makes it competitive with C4.5 on noisy data. Before pruning, C4.5 outperformed LSID3 on the *Monks-3* problem, which is known to be noisy. However, LSID3 was improved by pruning, eliminating the advantage C4.5 had. For some data sets, the trees induced by C4.5 are smaller than those learned by LSID3-p. However, the results indicate that among the 21 tasks for which t-test is applicable,[14] LSID3-p was significantly more accurate than C4.5 on 11, significantly worse on 2, and similar on the remaining 8. Taking into account all 24 data sets, the overall improvement by LSID3-p over C4.5 was found to be statistically significant by a Wilcoxon test with $\alpha = 0.05$. In general, LSID3-p performed as well as LSID3 on most data sets, and significantly better on the noisy ones.

14. The t-test is not applicable for the Monk data sets because only 1 train-test partition was used.

These results confirm our expectations: the problems addressed by LSID3 and C4.5's pruning are different. While LSID3 allots more time for better learning of hard concepts, pruning attempts to simplify the induced trees to avoid overfitting the data. The combination of LSID3 and pruning is shown to be worthwhile: it enjoys the benefits associated with lookahead without the need to compromise when the training set is noisy.

We also examined the effect of applying error-based pruning not only to the final tree, but to the lookahead trees as well. The experiments conducted on several noisy data sets showed that the results of this extra pruning phase were very similar to the results without pruning. Although pruning results in samples that better represent the final trees, it affects all samples similarly and hence does not lead to different split decisions.

4.2.3 BINARY SPLITS

By default, LSID3 uses multiway splits, that is, it builds a subtree for each possible value of a nominal attribute. Following the discussion in Section 2.5, we also tested how LSID3 performs if binary splits are forced. The tests in this case are found using exhaustive search.

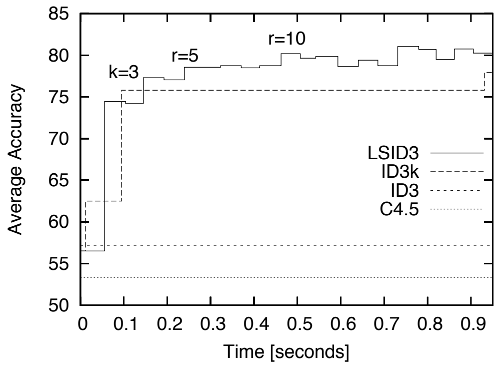
To demonstrate the fragmentation problem, we used two data sets. The first data set is *Tic-tac-toe*. When binary splits were forced, the performance of both C4.5 and LSID3-p improved from 85.8 and 87.2 to 94.1 and 94.8 respectively. As in the case of multiway splits, the advantage of LSID3-p over C4.5 is statistically significant with $\alpha = 0.05$. Note that binary splits come at a price: the number of candidate splits increases and the runtime becomes significantly longer. When allocated the same time budget, LSID3-p can afford larger samples than BLSID3-p. The advantage of the latter, however, is kept.

The second task is a variant on *XOR-2*, where there are 3 attributes a_1, a_2, a_3, each of which can take one of the values A, C, G, T. The target concept is $a_1^* \oplus a_2^*$. The values of a_i^* are obtained from a_i by mapping each A and C to 0 and each G and T to 1. The data set, referred to as *Categorial XOR-2*, consists of 16 randomly drawn examples. With multiway splits, both LSID3-p and C4.5 could not learn *Categorial XOR-2* and their accuracy was about 50%. C4.5 failed also with binary splits. BLSID3-p, on the contrary, was 92% accurate.

We also examined the bias of LSID3 toward binary attributes. For this purpose we used the example described in Section 2.5. An artificial data set with all possible values was created. LSID3 with multiway and with binary splits were run 10000 times. Although a_4 is as important to the class as a_1 and a_2, LSID3 with multiway splits never chose it at the root. LSID3 with binary splits, however, split the root on a_4 35% of the time. These results were similar to a_1 (33%) and a_2 (32%). They indicate that forcing binary splits removes the LSID3 bias.

4.3 Anytime Behavior of the Contract Algorithms

Both LSID3 and ID3-k make use of additional resources for generating better trees. However, in order to serve as good anytime algorithms, the quality of their output should improve with the increase in their allocated resources. For a typical anytime algorithm, this improvement is greater at the beginning and diminishes over time. To test the anytime behavior of LSID3 and ID3-k, we invoked them with successive values of r and k respectively. In the first set of experiments we focused on domains with nominal attributes, while in the second set we tested the anytime behavior for domains with continuous attributes.



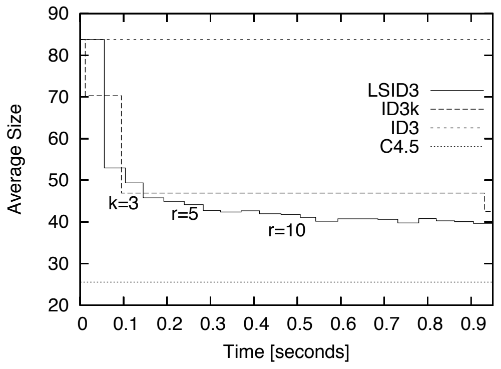
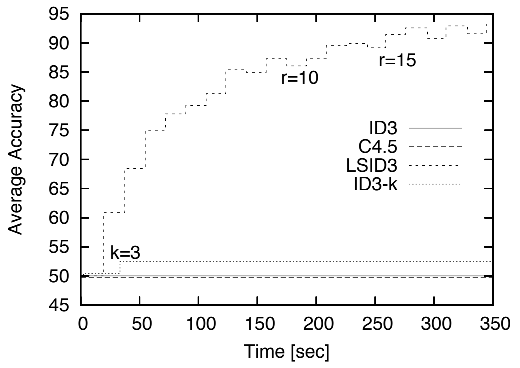


Figure 18: Anytime behavior of ID3-k and LSID3 on the *Multiplex-XOR* data set



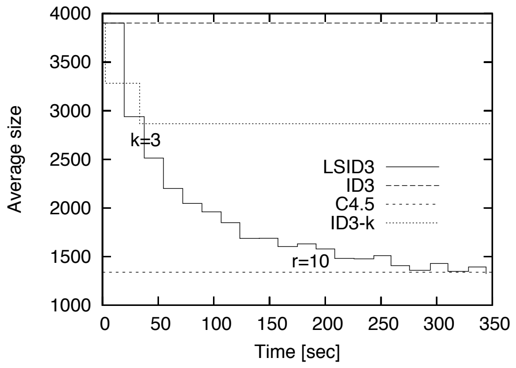


Figure 19: Anytime behavior of ID3-k and LSID3 on the *10-XOR* data set

4.3.1 NOMINAL ATTRIBUTES

Figures 18, 19 and 20 show the average results over 10 runs of 10-fold cross-validation experiments for the *Multiplex-XOR*, *XOR-10* and *Tic-tac-toe* data sets respectively. The x-axis represents the run time in seconds.[15] ID3 and C4.5, which are constant time algorithms, terminate quickly and do not improve with time. Since ID3-k with $k = 1$ and LSID3 with $r = 0$ are defined to be identical to ID3, the point at which ID3 yields a result serves also as the starting point of these anytime algorithms.

The graphs indicate that the anytime behavior of LSID3 is better than that of ID3-k. For ID3-k, the gaps between the points (width of the steps) increase exponentially, although successive values of k were used. As a result, any extra time budget that falls into one of these gaps cannot be exploited. For example, when run on the *XOR-10* data set, ID3-k is unable to make use of additional time that is longer than 33 seconds $(k = 3)$ but shorter than 350 seconds $(k = 4)$. For LSID3, the difference in the time required by the algorithm for any 2 successive values of r is almost the same.

For the *Multiplex-XOR* data set, the tree size and generalization accuracy improve with time for both LSID3 and ID3-k, and the improvement decreases with time. Except for a short period of time, LSID3 dominates ID3-k. For the *XOR-10* data set, LSID3 has a great advantage: while ID3-k

15. The algorithms were implemented in C++, compiled with GCC, and run on Macintosh G5 2.5 GHz.

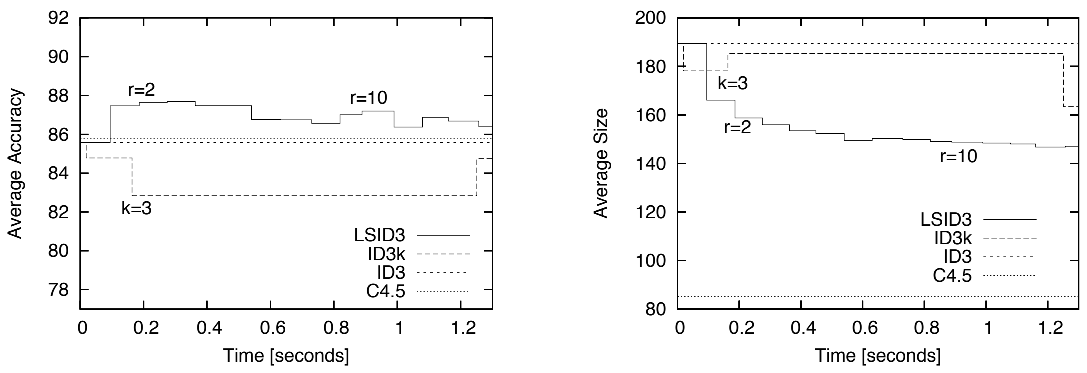


Figure 20: Anytime behavior of ID3-k and LSID3 on the *Tic-tac-toe* data set

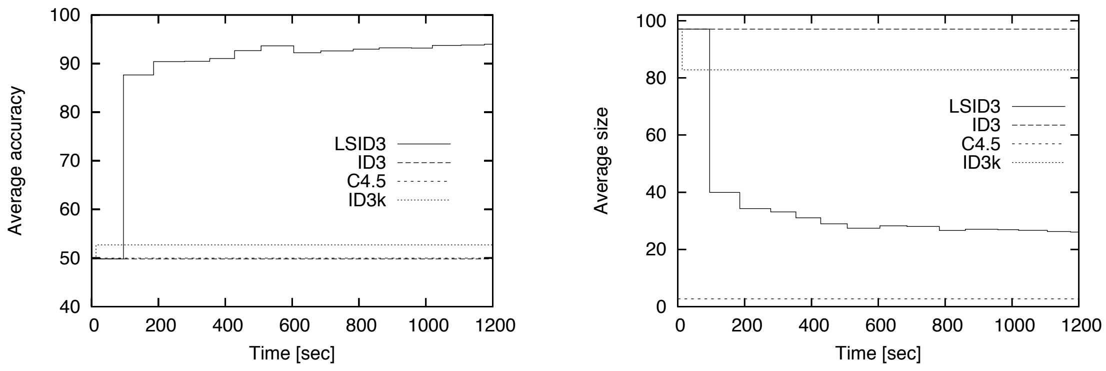


Figure 21: Anytime behavior of ID3-k, LSID3 on the *Numeric-XOR 4D* data set

produced trees whose accuracy was limited to 55%, LSID3 reached an average accuracy of more than 90%.

In the experiment with the *Tic-tac-toe* data set, LSID3 dominated ID3-k consistently, both in terms of accuracy and size. ID3-k performs poorly in this case. In addition to the large gaps between successive possible time allocations, a decrease in accuracy and an increase in tree size are observed at $k = 3$. Similar cases of pathology caused by limited-depth lookahead have been reported by Murthy and Salzberg (1995). Starting from $r = 5$, the accuracy of LSID3 does not improve over time and sometimes slightly declines (but still dominates ID3-k). We believe that the multiway splits prevent LSID3 from further improvements. Indeed, our experiments in Section 4.2 indicate that LSID3 can perform much better with binary splits.

4.3.2 CONTINUOUS ATTRIBUTES

Our next anytime-behavior experiment uses the *Numeric-XOR 4D* data set with continuous attributes. Figure 21 gives the results for ID3, C4.5, LSID3, and ID3-k. LSID3 clearly outperforms all the other algorithms and exhibits good anytime behavior. Generalization accuracy and tree size both improve with time. ID3-k behaves poorly in this case. For example, when 200 seconds are allocated, we can run LSID3 with $r = 2$ and achieve accuracy of about 90%. With the same allocation, ID3-k can be run with $k = 2$ and achieve accuracy of about 52%. The next improvement of

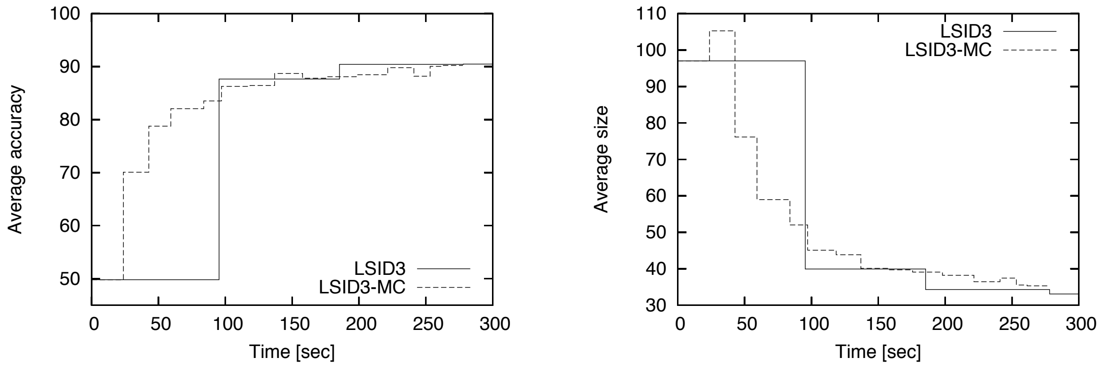


Figure 22: Anytime behavior of LSID3-MC on the *Numeric-XOR 4D* data set

ID3-k (with $k = 3$) requires 10,000 seconds. But even with such a large allocation (not shown in the graph since it is off the scale), the resulting accuracy is only about 66%.

In Section 2.4 we described the LSID3-MC algorithm which, instead of uniformly distributing evaluation resources over all possible splitting points, performs biased sampling towards points with high information gain. Figure 22 compares the anytime behavior of LSID3-MC to that of LSID3. The graph of LSID3 shows, as before, the performance for successive values of r. The graph of LSID3 shows the performance for $p = 10\%, 20\%, \ldots, 150\%$. A few significant conclusions can be drawn from these results:

1. The correlation between the parameter p and the runtime is almost linear: the steps in the graph are of almost constant duration.[16] We can easily increase the granularity of the anytime graph by smaller gaps between the p values.

2. It looks as if the runtime for LSID3-MC with $p = 100\%$ should be the same as LSID3($r = 1$) without sampling where all candidates are evaluated once. We can see, however, that the runtime of LSID3($r = 1$) is not sufficient for running LSID3-MC with $p = 50\%$. This is due to the overhead associated with the process of ordering the candidates prior to sample selection.

3. LSID3-MC with $p = 100\%$ performs better than LSID3($r = 1$). This difference can be explained by the fact that we performed a biased sample with repetitions, and therefore more resources were devoted to more promising tests rather than one repetition for each point as in LSID3($r = 1$).

4. When the available time is insufficient for running LSID3($r = 1$) but more than sufficient for running ID3, LSID3-MC is more flexible and allows these intermediate points of time to be exploited. For instance, by using only one-fifth of the time required by LSID3($r = 1$), an absolute accuracy improvement of 20% over ID3 was achieved.

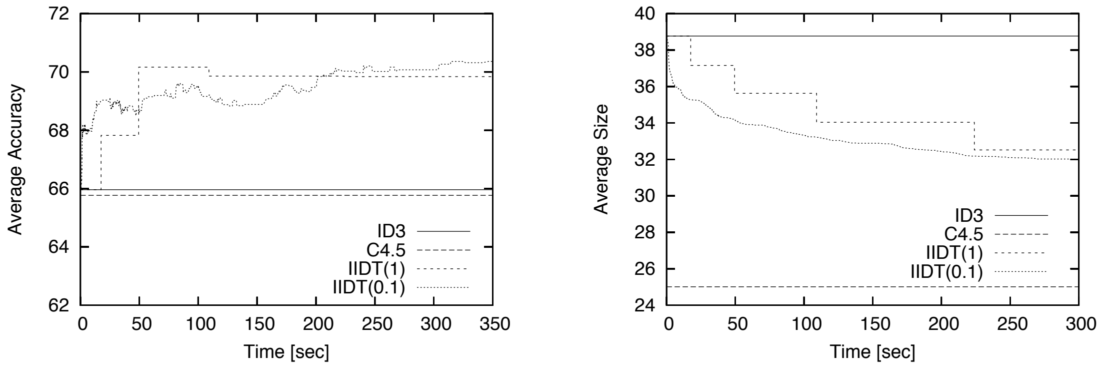


Figure 23: Anytime behavior of IIDT on the *Glass* data set

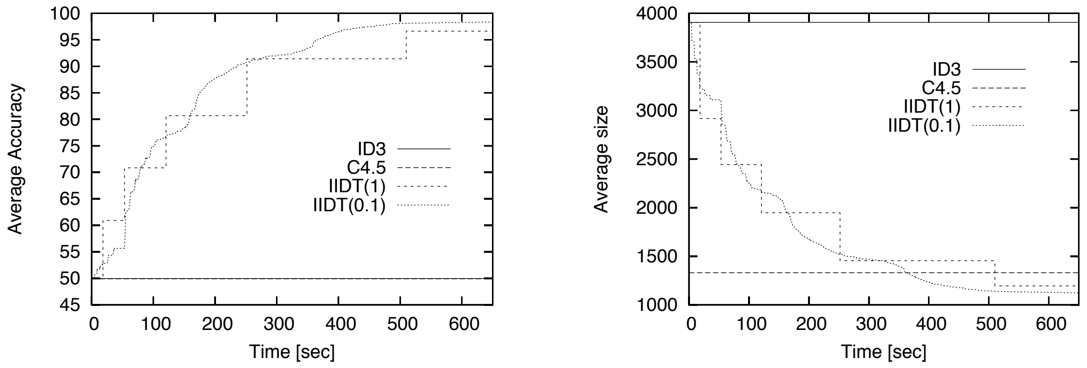


Figure 24: Anytime behavior of IIDT on the *10-XOR* data set

4.4 Anytime Behavior of IIDT

IIDT was presented as an interruptible decision tree learner that does not require advanced knowledge of its resource allocation: it can be stopped at any moment and return a valid decision tree. We tested two versions of IIDT, the first with granularity threshold $g = 0.1$ and the second with $g = 1$. Figures 23, 24, and 25 show the anytime performance of IIDT in terms of tree size and accuracy for the *Glass*, *XOR-10*, and *Tic-tac-toe* data sets. Each graph represents an average of 100 runs (for the 10×10 cross-validation). Unlike the graphs given in the previous section, these are interruptible anytime graphs, that is, for each point, the y coordinate reflects the performance if the algorithm was interrupted at the associated x coordinate. In the contract algorithm graphs, however, each point reflects the performance if the algorithm was initially allocated the time represented by the x coordinate.

In all cases, the two anytime versions indeed exploit the additional resources and produce both smaller and more accurate trees. Since our algorithm replaces a subtree only if the new one is smaller, all size graphs decrease monotonically. The most interesting anytime behavior is for the difficult *XOR-10* problem. There, the tree size decreases from 4000 leaves to almost the optimal size (1024), and the accuracy increases from 50% (which is the accuracy achieved by ID3 and C4.5)

16. Some steps look as though they require durations that are twice as long. However, these durations actually represent two p values with identical results.

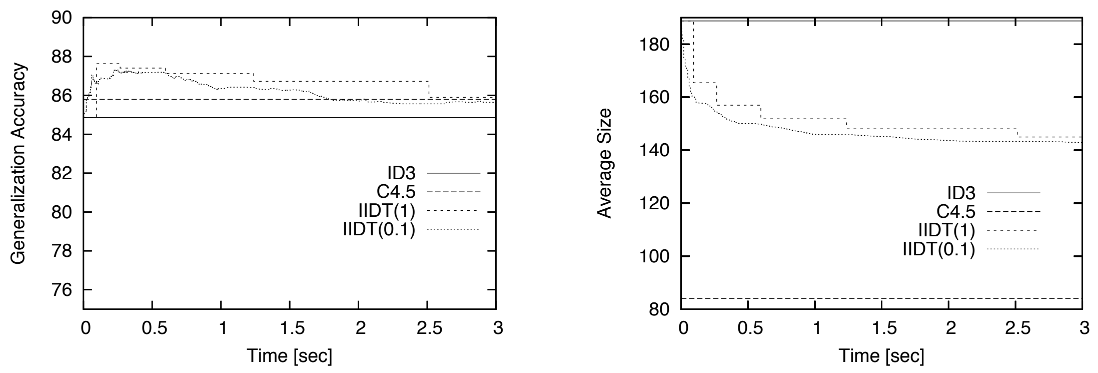


Figure 25: Anytime behavior of IIDT on the *Tic-tac-toe* data set

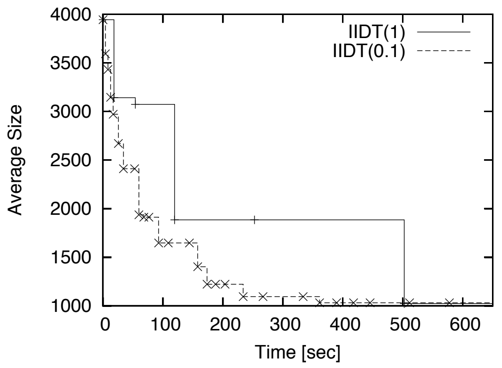


Figure 26: Time steps for a single run on *10-XOR*

to almost 100%. The shape of the graphs is typical to those of anytime algorithms with diminishing returns. The improvement in the accuracy of IIDT (at the latest point it was measured) over ID3 and C4.5 was found by t-test ($\alpha = 0.05$) to be significant for the *Glass* and *XOR-10* data sets. The performance of IIDT on *Tic-tac-toe* slightly degrades over time. We believe that similarly to LSID3, IIDT can perform much better if binary splits are used.

The difference in performance of the two anytime algorithms is interesting. IIDT(0.1), with the lower granularity parameter, indeed produces smoother anytime graphs (with lower volatility), which allows for better control and better predictability of return. Moreover, in large portions of the time axis, the IIDT(0.1) graph shows better performance than that of IIDT(1). This is due to more sophisticated node selection in the former. Recall that $g = 1$ means that the algorithm always selects the entire tree for improvement.

The smoothness of the IIDT(0.1) graphs is somehow misleading because it represents an average of 100 runs, with each step taking place at a different time (this is in contrast to the graph for IIDT(1), where the steps are at roughly the same times). Figure 26 illustrates the significance of this smoothing effect on a single anytime graph (out of the 100). We can see that although the IIDT(0.1) graph is less smooth than the average graph, it is still much smoother than the corresponding IIDT(1) graph.

4.5 Comparison with Modern Decision Tree Learners

During the last decade, several modern decision tree learners were introduced. Although these learners were not presented and studied as anytime algorithms, some of them can be viewed as such. In what follows we compare our proposed anytime framework to three such algorithms: bagging, skewing and GATree. We first give a brief overview of the studied methods and then compare their performance to that of our anytime approach.

4.5.1 OVERVIEW OF THE COMPARED METHODS

Page and Ray (2003) introduced *skewing* as an alternative to lookahead for addressing problematic concepts such as parity functions. At each node, the algorithm skews the set of examples and produces several versions of it, each with different weights for the instances. The algorithm chooses to split on the attribute that exceeds a pre-set gain threshold for the greatest number of weightings. Skewing was reported to perform well on hard concepts such as XOR-n, mainly when the data set is large enough relative to the number of attributes. Skewing can be viewed as a contract algorithm parameterized by w, the number of weightings. In order to convert skewing into an interruptible algorithm, we apply the general conversion method described in Section 3.1.

Two improvements to the original skewing algorithm were presented, namely *sequential skewing* (Page and Ray, 2004), which skews one variable at a time instead of all of them simultaneously, and *generalized skewing* (Ray and Page, 2005), which can handle nominal and continuous attributes. Nominal attributes are skewed by randomly reordering the possible values and assigning a weight for a value proportional to its rank. Continuous attributes are handled by altering the input distribution for every possible split point. We test the sequential skewing algorithm on the binary *XOR* and *Multiplexer* data sets. This version is not parameterized and hence is not anytime by nature. To convert it into an anytime algorithm, we added a parameter k that controls the number of skewing iterations. Thus, instead of skewing each variable once, we skew it k times. For the *Tic-tac-toe* data set, where the attributes are ternary, we used only the generalized skewing algorithm, parameterized by the number of random orderings by which the nominal attributes are reweighed.

Papagelis and Kalles (2001) presented *GATree*, a learner that uses genetic algorithms to evolve decision trees. The initial population consists of randomly generated 1-level depth decision trees, where both the test and the class labels are drawn randomly. Mutation steps choose a random node and replaces its test with a new random one. If the chosen node is a leaf, the class label is replaced by a random label. Crossover chooses two random nodes, possibly from different trees, and swaps their subtrees. When tested on several UCI data sets, GATree was reported to produce trees as accurate as C4.5 but of significantly smaller size. GATree was also shown to outperform C4.5 on the *XOR-2* and *XOR-3* problems. GATree can be viewed as an interruptible anytime algorithm that uses additional time to produce more and more generations. In our experiments we used the GATree full original version with the same set of parameters as reported by Papagelis and Kalles, with one exception: we allowed a larger number of generations.

The third algorithm we tested is *bagging* (Breiman, 1996). Bagging is an ensemble-based method, and as such, it is naturally an interruptible anytime learner. Additional resources can be exploited by bagging to generate larger committees. In our experiments we consider 3 different bagging methods that use ID3, C4.5, and RTG (Random Tree Generator) as base learners. In addition, we tested a committee of trees produced by our LSID3. Since the latter takes significantly more time to run, the LSID3 committees are expected to be smaller than the greedy committees for

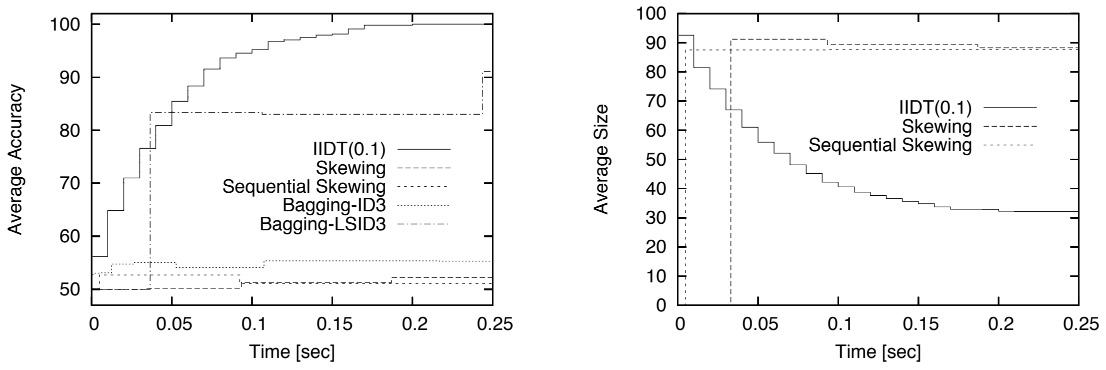


Figure 27: Anytime behavior of various modern algorithms on the *XOR-5* data set

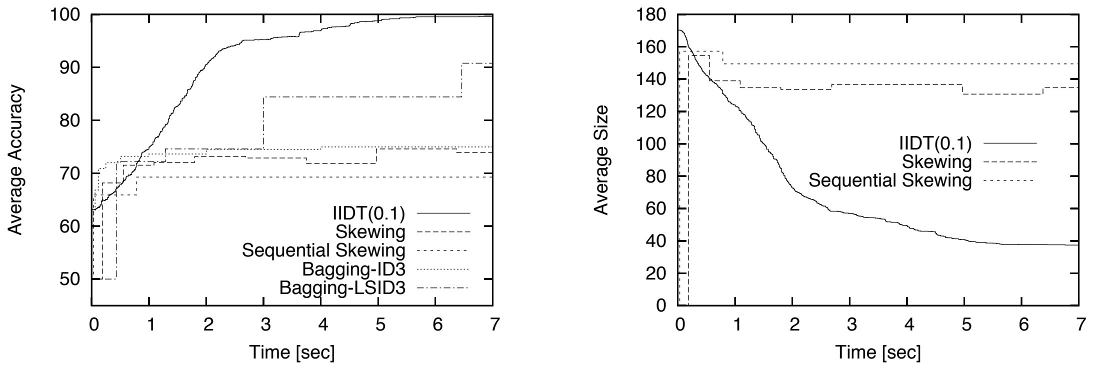


Figure 28: Anytime behavior of various modern algorithms on the *Multiplexer-20* data set

the same allocated time. Note that bagging is not a direct competitor to our method. We defined our goal as inducing a single "good" decision tree while bagging generates a set of trees. Generating a set of trees rather than a single good tree eliminates one of the greatest advantages of decision trees—their comprehensibility.

4.5.2 EMPIRICAL COMPARISON

We used our own implementation for IIDT, skewing, and bagging, and the commercial version for GATree.[17] The skewing and sequential skewing versions were run with linearly increasing parameters. The generalized skewing algorithm was run with exponentially increasing parameters. The performance of the ensemble method was tested for exponentially increasing committee sizes $(1,2,4,8,\ldots)$.

Figures 27, 28, and 29 compare IIDT to bagging with ID3 as a base learner, bagging with LSID3($r=1$), and skewing on the *XOR-5*, *Multiplexer-20*, and *Tic-tac-toe* tasks respectively. Note that the results for ID3 are identical to those of bagging-ID3 with a single tree in the committee and

17. The experiments with GATree were run on a Pentium IV 2.8 GHz machine with the Windows XP operating system. The reported times are as output by the application itself.

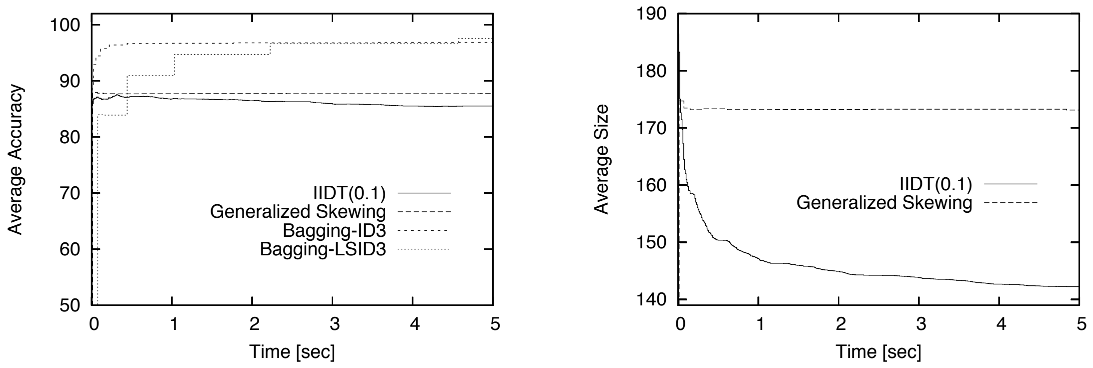


Figure 29: Anytime behavior of various modern algorithms on the *Tic-tac-toe* data set

hence are not plotted independently. Since GATree was run on a different machine, we report its results separately later in this section.

The graphs for the first 2 problems, which are known to be hard, show that IIDT clearly outperforms the other methods both in terms of tree size and accuracy. In both cases IIDT reaches almost perfect accuracy (99%), while bagging-ID3 and skewing topped at 55% for the first problem and 75% for the second.

The inferior performance of bagging-ID3 on the *XOR-5* and *Multiplexer-20* tasks is not surprising. The trees that form the committee were induced greedily and hence could not discover these difficult concepts, even when they were combined. Similar results were obtained when running bagging over C4.5 and RTG. However, when our LSID3($r = 1$) was used as a base learner, performance was significantly better than that of the greedy committees. Still, IIDT performed significantly better than bagging-LSID3, indicating that for difficult concepts, it is better to invest more resources for improving a single tree than for adding more trees of lower quality to the committee.

The inferior results of the skewing algorithms are more difficult to interpret, since skewing was shown to handle difficult concepts well. One possible explanation for this is the small number of examples with respect to the difficulty of the problem. To verify that this indeed explains the inferior results, we repeated the experiment with simpler XOR problems such as XOR-2 and XOR-3. In these cases skewing indeed did much better and outperformed ID3, reaching 100% accuracy (as IIDT). When we increased the size of the training set for the XOR-5 domain, skewing also performed better, yet IIDT outperformed it by more than 9%. For a deeper analysis of the difference between IIDT and skewing, see Section 5.

The average accuracy of GATree, after 150 generations, was 49.5%. It took more than 3 seconds on average to reach 150 generations. Thus, even when GATree was allocated much more time than IIDT, it could not compete with the latter. We repeated the experiment, allowing GATree to have a larger initial population and to produce more generations. The accuracy on the testing set, even after thousands of generations, remained very low. Similar results were obtained for the *Multiplexers-20* data set.

The above set of experiments was repeated on the much more difficult *XOR-10* data set. The advantage of IIDT over the other methods was even more evident. While IIDT was able to reach accuracy of 100%, bagging-ID3, skewing, and GATree performed as poorly as a random guesser, with accuracy of only 50%.

The next experiment was with the *Tic-tac-toe* data set. In this case, as shown in Figure 29, both ensemble-based methods have a significant advantage over the single tree inducers. We speculate that this is because ensemble methods were able to overcome the quick-fragmentation problem associated with multiway splits by combining several classifiers. We are still looking for ways to verify this hypothesis. Bagging-ID3 outperforms the other methods until the fifth second, where bagging-LSID3 overtakes it slightly. In contrast to the *XOR-5* domain, building larger committees is worthwhile in this case, even at the expense of less accurate base classifiers. However, if the time allocation permits, large ensembles of LSID3 trees are shown to be the most accurate. We believe that the general question of tradeoff between the resources allocated for each tree and the number of trees forming the ensemble should be addressed by further research with extensive experiments on various data sets. The performance of generalized skewing and IIDT was similar in this case, with a slight advantage for skewing in terms of accuracy and an advantage for IIDT in terms of tree size. GATree was run on the data set for 150 generations (30 seconds). The average accuracy was 76.42%, much lower than that of the other methods.

5. Related Work

While to the best of our knowledge no other work has tried to design an anytime algorithm for decision tree induction in particular, there are several related works that warrant discussion here. We consider first works that deal with decision tree inducers. We then discuss algorithms for induction of other models, and finally we consider methods that focus on aspects of the learning process other than the model-induction phase.

5.1 Single Tree Inducers

The goal of our research was to develop anytime algorithms for inducing a single decision tree. Several other algorithms for single decision-tree induction can either be considered anytime algorithms or can be converted into them with relative ease. We view the recently introduced skewing approach as the most relevant to our work. Therefore, we focus first on the differences between skewing and our anytime framework, and then we consider other decision tree learners.

5.1.1 SKEWING

The skewing approach was presented as an efficient alternative to lookahead (Page and Ray, 2003). In our discussion, we analyze the original skewing algorithm, which is applicable only to binary attributes. This analysis, however, holds also for the other versions of skewing (see Section 4.5), since they are based on the same ideas.

Skewing relies on the hypothesis that, when learning a hard concept, it may be easier for a greedy decision tree inducer to pick a relevant attribute if the distribution over the data is significantly different from the uniform distribution. Therefore, at each node, skewing repeatedly attempts to produce, from the existing set of examples, skewed versions that represent different distributions. Several different weight vectors are repeatedly assigned to the examples for this purpose. Then, the information gain of each attribute is calculated on the basis of each weight vector. The attribute that exceeds a pre-determined gain threshold for the greatest number of times is chosen. In order to generate the weight vectors, a *favored value* v_i is randomly drawn for each candidate attribute a_i. Then, the weight of each example in which a_i takes the value v_i is increased.

This process can be viewed as a stochastic clustering of the examples according to the values they take for a subset of the attributes: examples that most agree with the favored value for a large number of attributes are assigned the highest weight. Therefore, the calculation of the information gain will be most affected by sets of examples that share the same values for different subsets of the attributes.

In regular k-steps lookahead, for each tuple of k attributes we divide the set of examples into 2^k subsets, each of which is associated with a different k-tuple of binary values, that is, a different sub-path of length k in the lookahead tree. Then, we calculate the information gain for each subset and compute the average gain weighted by subset sizes.

In skewing, each iteration assigns a high weight to a few subsets of the examples that have common values for some of the attributes. These subsets have the greatest influence on the gain calculation at that skewing iteration. Hence, skewing iterations can be seen as sampling the space of lookahead sub-paths and considering a few of them at a time.

The great advantage of skewing over our proposed framework is efficiency. While LSID3 samples full decision trees, skewing samples only single sub-paths. When there are many examples but relatively few attributes, skewing improves greatly over the greedy learners at a low cost. The effectiveness of skewing, however, noticeably degrades when the concept hides a mutual dependency between a large number of attributes. When the number of attributes increases, it becomes harder to create large clusters with common values for some of the relevant ones, and hence the resulting lookahead is shallower.

Consider, for example, the *n-XOR* problem with n additional irrelevant attributes. For $n = 2$, a reweighting that assigns a high weight to examples that agree only on 1 of the 2 relevant attributes results in a high gain for the second attribute because it decreases the entropy of the cluster to 0. Nevertheless, in order to have a positive gain for a relevant attribute when $n = 10$, we should cluster together examples that agree on 9 of the 10 relevant attributes. Obviously, the probability for a good cluster in the first case is high while in the second case it is almost 0. The experiments reported in Section 4.5 provide an empirical backup for this argument.

5.1.2 OTHER SINGLE TREE INDUCERS

Papagelis and Kalles (2001) studied GATree, a learner that uses genetic algorithms for building decision trees. GATree does not adopt the top-down scheme. Instead, it starts with a population of random trees and uses a mutation operation of randomly changing a splitting test and a crossover operation of exchanging subtrees. Unlike our approach, GATree is not designed to generate consistent decision trees and searches the space of all possible trees over a given set of attributes. Thus, it is not appropriate for applications where a consistent tree is required. Like most genetic algorithms, GATree requires cautious parameter tuning and its performance depends greatly on the chosen setting. Comparing GATree to our algorithm (see Section 4.5) shows that, especially for hard concepts, it is much better to invest the resources in careful tuning of a single tree than to perform genetic search over the large population of decision trees.

Utgoff et al. (1997) presented DMTI, an induction algorithm that chooses an attribute by building a single decision tree under each candidate attribute and evaluates it using various measures. Several possible tree measures were examined and the MDL (Minimum Description Length) measure performed best. DMTI is similar to LSID3($r = 1$) but, unlike LSID3, it can only use a fixed amount of additional resources and hence cannot serve as an anytime algorithm. When the user can afford using more resources than required by DMTI, the latter does not provide means to improve the learned model further. Furthermore, DMTI uses a single greedy lookahead tree for attribute evaluation, while we use a biased sample of the possible lookahead trees. Our experiments with DMTI (as available online) show that while it can solve simpler XOR and multiplexer problems, its limited lookahead is not sufficient for learning complex concepts such as XOR-10: DMTI achieved an accuracy of 50%. IIDT and LSID3, by producing larger samples, overcame this problem and reached high accuracies.

Kim and Loh (2001) introduced CRUISE, a bias-free decision tree learner that attempts to produce more compact trees by (1) using multiway splits—one subnode for each class, and (2) examining pair-wise interactions among the variables. CRUISE is able to learn XOR-2 and Chess-board (numeric XOR-2) concepts. Much like ID3-k with $k = 2$, it cannot recognize more complex interactions.

Bennett (1994) presented GTO, a non-greedy approach for repairing multivariate decision trees. GTO requires as input an initial tree. The algorithm retains the structure of the tree but attempts to simultaneously improve all the multivariate decisions of the tree using iterative linear programming. GTO and IIDT both use a non-greedy approach to improve a decision tree. The advantage of GTO is its use of a well-established numerical method for optimization. Its disadvantages are its inability to modify the initial structure and its inability to exploit additional resources (beyond those needed for convergence).

5.2 Induction of Other Models

Ensemble-based methods, such as bagging and boosting (Schapire, 1999), can also be viewed as anytime algorithms. However, the classifiers constructed by the bagging and boosting algorithms consist of a committee of decision trees rather than a single tree. Therefore, the problem they face is very different from the problem we face in this work—that of learning a single tree. A major problem with ensemble-based methods is that the induced ensemble is often large, complex, and difficult to interpret (Freund and Mason, 1999). Therefore, these methods cannot be used when comprehensible models are required. Another problem is that greedy trees are unable to discover any knowledge about hard-to-learn target concepts. Therefore, combining them cannot improve performance. In our experiments, reported in Section 4.5, we provide empirical support for this claim by comparing our proposed anytime algorithms to bagging.

Dietterich (2000) presented the randomized-C4.5 algorithm, where a randomized version of C4.5 that chooses the split attribute at random from among the 20 best candidates is repeatedly invoked to produce an ensemble of decision trees. The experimental results indicate that ensembles of randomized-C4.5 were competitive with bagging but not as accurate as boosting. As in the case of the traditional bagging and boosting methods, our framework differs from randomized-C4.5 in that the latter produces an ensemble of trees that is obviously not as comprehensible as a single decision trees is.

The SVM algorithm usually depends on several parameters—kernel parameters for example. Several works, such as Chapelle et al. (2002), proposed iterative methods for automatic tuning of SVM parameters. These iterative methods can exploit additional time resources for better tuning.

Opitz (1995) introduced an anytime approach for theory refinement. This approach starts by generating a neural network from a set of rules that describe what is currently known about the domain. The network then uses the training data and the additional time resources to try to improve the resulting hypothesis.

5.3 Other Learning Components

In addition to the induction procedure, the learning process involves other components, where additional resources can be invested. Many algorithms for active learning (Lindenbaum et al., 2004), feature generation (Markovitch and Rosenstein, 2002), and feature selection (Last et al., 2001) have some level of anytime characteristics. All these methods are orthogonal to our induction methods and can complement them. An interesting research direction is to determine resource distribution between the various learning stages.

The related problem of *budgeted learning* was defined by Lizotte et al. (2003). There is a cost associated with obtaining each attribute value of a training example, and the task is to determine what attributes to test given a budget.

6. Conclusions

With the increased popularity of Machine Learning techniques, induction algorithms are being applied to more complex problems. Recently, much research effort has been spent on developing advanced induction algorithms, such as SVM. However, these algorithms do not provide comprehensible models and therefore are not appropriate when understandability is crucial, such as in medical tasks. Decision trees, on the contrary, are considered easy to interpret. In order to allow induction of better decision trees for hard-to-learn concepts, we presented a framework for anytime induction of decision trees that makes use of additional resources for producing better hypotheses.

Existing greedy algorithms for learning decision trees require a fixed small amount of time. We showed that for several real-world and synthetic data sets, our proposed anytime algorithms can exploit larger time budgets. This is shown to be worthwhile especially for hard concepts where existing greedy methods fail.

The major contributions of this paper are:

- *A better understanding of the space of consistent decision trees:* Using sampling techniques, we conducted experiments that support the application of Occam's Razor for decision trees and showed that smaller trees are clearly preferable.

- *LSID3:* We presented and empirically evaluated LSID3, a contract anytime algorithm for learning decision trees. On the data sets studied in this paper, LSID3 was shown to exhibit good anytime behavior and to produce better decision trees.

- *IIDT:* Motivated by the need for interruptible learners, we introduced IIDT, an interruptible anytime algorithm for inducing decision trees. In our experiments on hard concepts, IIDT produced significantly better decision trees when run for longer time.

- *Comparison with modern learners:* Our proposed framework was compared to several modern decision tree learners and shown to outperform them significantly. The advantage of LSID3 and IIDT is most evident when applied on problems with a strong interdependency.

When one can afford allocating extra resources, both LSID3 and IIDT are considered among the best choices for inducing a decision tree.

To the best of our knowledge, this is the first work that proposes anytime algorithms for learning decision trees. This research is only a first step in this direction. We are currently in the process of designing and implementing many variations of our proposed algorithms, including algorithms that combine ID3-k and LSID3, algorithms that allow different resource distribution for different sub-concepts, and algorithms that allow pruning as an improvement phase in the interruptible learner. In addition, we intend to adapt our anytime framework for incremental tasks, where new examples arrive after the learning process has been initiated. This will allow the classifier to exploit both additional time resources and additional data that becomes available with time.

Acknowledgments

This work was partially supported by funding from the EC-sponsored MUSCLE Network of Excellence (FP6-507752).

Appendix A. Data Sets

Below we give a more detailed description of the data sets used in our experiments:

1. *Automobile:* This problem, taken from the UCI Repository, consists of three types of entities: (1) the specification of an automobile, (2) its assigned insurance risk rating, and (3) its normalized losses in use as compared to other cars. Several of the attributes in the database could be used as class attributes: we chose to use both the make (model) and the symbolling (risk degree).

2. *Balance Scale:* This data set is taken from the UCI Repository and was generated to model psychological experimental results. Each example is classified as having the balance scale tip to the right, tip to the left, or be balanced. The attributes are the left weight, the left distance, the right weight and the right distance.

3. *Breast Cancer (Ljubljana):* This problem is taken from the UCI Repository. Each instance represents the characteristics of a patient and the class is whether or not there are recurrence events.

4. *Connect-4:* This data set, taken from the UCI Repository, contains all legal 8-ply positions in the connect-4 game in which neither player has won yet and the next move is not forced.

5. *Corral:* An artificial data set first used by John et al. (1994).

6. *Glass:* In this domain, taken from the UCI Repository, the goal is to determine the type of glass from its characteristics.

7. *Iris:* This data set is taken from the UCI Repository and contains 3 classes of 50 instances each, where each class refers to a type of iris plant.

8. *Monks problems:* This set, taken from the UCI Repository, contains three problems. Each example is represented by 5 nominal attributes in the range $1,2,3,4$. The problems are:

   - *Monks-1:* $(a1 = a2) or (a5 = 1)$.
   - *Monks-2:* exactly two of $(a1 = 1, a2 = 1, a3 = 1, a4 = 1, a5 = 1)$.
   - *Monks-3:* $((a5 = 3) and (a4 = 1)) or ((a5 \neq 4) and (a2 \neq 3))$, with an added 5% class noise.

   The original data sets are already partitioned into training and testing sets.

9. *Mushroom:* This data set, taken from the UCI Repository, includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota family. Each species is identified as edible or poisonous.

10. *Solar Flare:* This problem is taken from the UCI Repository. Each instance represents captured features for one active region on the sun. Among the three possible classes, we considered the C-class flare where the instances are more distributed.

11. *Tic-Tac-Toe:* The problem, taken from the UCI Repository, deals with the classification of legal tic-tac-toe end games, as wins or non-wins for the x player. Each example is represented by 9 nominal attributes, which represent the slot values.

12. *Voting:* This data set, taken from the UCI repository, includes votes for each member of the U.S. House of Representatives on the 16 key votes identified by the CQA. The class of each record is Democrat or Republican.

13. *Wine:* This problem, taken from the UCI Repository, deals with the classification of wines into 3 class types. Each example is represented by 13 continuous attributes, which represent measures of chemical elements in the wine.

14. *Zoo:* In this domain, taken from the UCI Repository, the goal is to determine the type of the animal from several attributes.

15. *Multiplexer:* The multiplexer task was used by several researchers for evaluating classifiers, for example, Quinlan (1993). An instance is a series of bits of length $a + 2^a$, where a is a positive integer. The first a bits represent an index into the remaining bits and the label of the instance is the value of the indexed bit. In our experiments we considered the 20-Multiplexer ($a = 4$). The data set contains 500 randomly drawn instances.

16. *Boolean XOR:* Parity-like functions are known to be problematic for many learning algorithms. However, they naturally arise in real-world data, such as the Drosophila survival concept (Page and Ray, 2003). We considered XOR of five and ten variables with additional irrelevant attributes.

17. *Numeric XOR:* A XOR based numeric data set that has been used to evaluate learning algorithms, for example, Baram et al. (2003). Each example consists of values for x and y coordinates. The example is labelled 1 if the product of x and y is positive, and -1 otherwise. We generalized this domain for three and four dimensions and added irrelevant variables to make the concept harder.

18. *Multiplex-XOR:* Another XOR based concept that is defined over 11 binary attributes. The concept is a composition of two XOR terms, where the first attribute determines which one should be considered. The other 10 attributes are used to form the XOR terms. The size of each term is drawn randomly between 2 and 5.

Appendix B. Full Results

Table 2 shows the size of the trees induced by the above mentioned algorithms, lists the differences between the algorithms, and states whether these differences were found to have t-test significance with $\alpha = 0.05$. Similarly, Table 3 compares the produced trees in terms of their generalization accuracy.

DATA SET	ID3	C4.5	ID3-K ($k=2$)	LSID3 ($r=5$)	LSID3-P ($r=5$)	LSID3 VS. ID3 DIFF	SIG?	LSID3 VS. C4.5 DIFF	SIG?	LSID3-P VS. C4.5 DIFF	SIG?
AUTOS MAKE	53.6 ±4.2	26.6 ±1.4	56.5 ±2.4	36.6 ±2.0	37.1 ±2.0	-17.1 ±4.8	√	9.9 ±2.4	×	10.4 ±2.2	×
AUTOS SYM.	46.7 ±1.8	20.1 ±4.1	47.0 ±1.8	26.8 ±1.7	26.5 ±2.1	-19.9 ±2.2	√	6.6 ±4.1	×	6.3 ±4.3	×
BALANCE	353.9 ±6.9	34.8 ±5.5	351.6 ±7.3	347.7 ±7.7	40.1 ±6.5	-6.2 ±5.0	√	312.8 ±9.8	×	5.2 ±7.7	×
BR. CANCER	129.6 ±5.9	6.1 ±4.0	124.0 ±4.9	100.3 ±4.4	7.7 ±7.1	-29.3 ±5.6	√	94.2 ±5.4	×	1.7 ±8.5	~
CONNECT-4	18507 ±139	3329 ±64	16143 ±44	14531 ±168	6614 ±173	-3976 ±265	√	11201 ±183	×	3284 ±186	×
CORRAL	9.2 ±2.1	5.3 ±1.2	7.0 ±0.6	6.7 ±0.5	6.3 ±0.8	-2.5 ±2.2	√	1.4 ±1.3	×	1.0 ±1.3	×
GLASS	38.7 ±2.3	23.9 ±2.6	32.3 ±2.3	34.2 ±2.1	35.5 ±2.4	-4.4 ±3.0	√	10.3 ±3.1	×	11.6 ±3.7	×
IRIS	8.5 ±1.0	4.7 ±0.5	9.1 ±0.9	7.6 ±0.8	6.6 ±1.6	-0.9 ±0.7	√	2.9 ±1.0	×	1.9 ±1.6	×
MONKS-1	62.0 ±0.0	11.0 ±0.0	27.0 ±0.0	27.0 ±0.0	21.0 ±0.0	-35.0 ±0.0	-	16.0 ±0.0	-	10.0 ±0.0	-
MONKS-2	109.0 ±0.0	20.0 ±0.0	105.0 ±0.0	92.4 ±3.4	18.3 ±5.0	-16.6 ±3.4	-	72.4 ±3.4	-	-1.7 ±5.0	-
MONKS-3	31.0 ±0.0	9.0 ±0.0	34.0 ±0.0	26.8 ±1.6	9.9 ±1.4	-4.2 ±1.6	-	17.8 ±1.6	-	0.9 ±1.4	-
MUSHROOM	24.0 ±0.0	19.0 ±0.2	24.9 ±0.2	16.2 ±0.9	16.2 ±0.9	-7.8 ±0.9	√	-2.8 ±0.9	√	-2.9 ±0.9	√
SOLAR-FLARE	68.9 ±2.9	2.0 ±1.0	68.4 ±3.2	63.2 ±2.9	1.2 ±0.8	-5.6 ±2.5	√	61.2 ±3.1	×	-0.8 ±1.4	√
TIC-TAC-TOE	189.0 ±13.6	83.4 ±7.8	176.7 ±9.0	151.7 ±5.6	112.6 ±8.9	-37.3 ±15.1	√	68.3 ±9.3	×	29.1 ±10.9	×
VOTING	13.6 ±2.2	2.8 ±1.5	12.3 ±1.6	13.0 ±2.0	3.5 ±2.1	-0.6 ±2.1	√	10.2 ±2.5	×	0.7 ±2.4	×
WINE	7.9 ±1.0	5.3 ±0.7	7.3 ±0.9	6.2 ±0.7	7.4 ±1.3	-1.7 ±1.2	√	0.9 ±1.0	×	2.1 ±1.6	×
ZOO	13.8 ±0.4	8.3 ±0.8	13.8 ±0.5	9.9 ±0.8	9.8 ±0.9	-3.9 ±0.9	√	1.6 ±1.2	×	1.5 ±1.2	×
NUMERIC XOR-3D	43.0 ±5.1	1.0 ±0.1	15.6 ±6.8	9.2 ±1.0	11.7 ±1.7	-33.8 ±5.1	√	8.2 ±1.0	×	10.7 ±1.7	×
NUMERIC XOR-4D	104.7 ±4.5	2.7 ±1.8	75.5 ±14.0	26.8 ±4.7	30.9 ±5.9	-77.9 ±6.0	√	24.1 ±4.8	×	28.1 ±6.3	×
MULTIPLEXER-20	142.8 ±8.3	66.1 ±6.5	67.1 ±29.0	46.6 ±20.0	41.2 ±16.8	-96.1 ±21.6	√	-19.4 ±20.4	√	-24.9 ±17.4	√
MULTIPLEX-XOR	84.0 ±5.6	26.0 ±5.2	70.2 ±5.3	43.9 ±5.7	31.6 ±4.8	-40.1 ±7.3	√	17.9 ±8.1	×	5.7 ±7.0	×
XOR-5	92.3 ±7.8	21.9 ±5.3	82.1 ±9.3	32.0 ±0.0	32.0 ±0.0	-60.3 ±7.8	√	10.1 ±5.3	×	10.1 ±5.3	×
XOR-5 NOISE	93.6 ±6.0	23.2 ±5.9	82.4 ±7.3	58.2 ±6.1	40.4 ±5.0	-35.4 ±8.3	√	35.0 ±8.9	×	17.2 ±8.1	×
XOR-10	3901 ±34	1367 ±39	3287 ±37	2004 ±585	1641 ±524	-1897 ±587	√	637 ±577	×	273 ±524	×

Table 2: The size of the induced trees on various data sets. The numbers represent the average and standard deviation over the individual runs. The last columns list the average differences between the algorithms and their t-test significance, with $\alpha = 0.05$ ($\surd$ indicates a significant advantage and $\times$ a significant disadvantage). The t-test is not applicable for the Monk data sets because only 1 train-test partition was used.

DATA SET	ID3	C4.5	ID3-K ($k=2$)	LSID3 ($r=5$)	LSID3-P ($r=5$)	LSID3 VS. ID3 DIFF	SIG?	LSID3 VS. C4.5 DIFF	SIG?	LSID3-P VS. C4.5 DIFF	SIG?
AUTOS MAKE	79.1 ±9.8	79.4 ±10.5	78.0 ±9.7	80.3 ±9.9	79.2 ±10.2	1.2 ±9.9	~	0.8 ±10.3	~	-0.2 ±10.6	~
AUTOS SYM.	81.9 ±9.8	77.4 ±10.6	81.2 ±9.5	81.3 ±10.2	81.1 ±9.5	-0.6 ±11.4	~	3.9 ±12.3	√	3.7 ±12.2	√
BALANCE	68.8 ±5.4	63.8 ±4.9	69.7 ±5.0	70.3 ±5.7	67.9 ±5.5	1.5 ±4.5	√	6.5 ±5.6	√	4.1 ±5.5	√
BR. CANCER	67.0 ±8.8	74.7 ±8.1	64.4 ±8.4	66.9 ±9.1	71.1 ±8.2	-0.2 ±9.1	~	-7.8 ±9.1	×	-3.6 ±6.0	×
CONNECT-4	75.6 ±0.5	81.3 ±0.4	77.9 ±0.5	78.6 ±0.5	80.4 ±0.5	3.0 ±0.6	√	-2.7 ±0.5	×	-0.9 ±0.4	×
CORRAL	69.7 ±25.4	65.3 ±19.8	89.0 ±19.9	83.5 ±20.9	79.6 ±19.8	13.8 ±25.4	√	18.2 ±25.1	√	14.2 ±24.1	√
GLASS	66.1 ±10.2	67.4 ±10.3	70.6 ±11.0	70.5 ±10.7	69.5 ±10.4	4.3 ±11.6	√	3.0 ±12.7	√	2.0 ±11.5	~
IRIS	93.3 ±5.9	95.3 ±4.8	93.1 ±6.7	94.5 ±6.0	94.3 ±6.2	1.1 ±4.0	√	-0.8 ±7.8	~	-0.9 ±7.9	~
MONKS-1	82.9 ±0.0	75.7 ±0.0	100.0 ±0.0	100.0 ±0.0	94.4 ±0.0	17.1 ±0.0	-	24.3 ±0.0	-	18.8 ±0.0	-
MONKS-2	69.2 ±0.0	65.0 ±0.0	63.0 ±0.0	67.1 ±0.5	63.6 ±1.1	-2.2 ±0.5	-	2.0 ±0.5	-	-1.5 ±1.1	-
MONKS-3	94.4 ±0.0	97.2 ±0.0	91.7 ±0.0	91.5 ±1.5	98.1 ±1.3	-3.0 ±1.5	-	-5.7 ±1.5	-	0.8 ±1.3	-
MUSHROOM	100.0 ±0.0	100.0 ±0.0	100.0 ±0.0	100.0 ±0.0	100.0 ±0.0	0.0 ±0.0	~	0.0 ±0.0	~	0.0 ±0.0	~
SOLAR FLARE	86.6 ±6.1	88.9 ±5.2	86.6 ±6.0	86.5 ±6.4	88.9 ±5.1	-0.2 ±3.0	~	-2.4 ±8.0	×	0.0 ±7.6	~
TIC-TAC-TOE	85.5 ±3.8	85.8 ±3.3	84.2 ±3.4	87.0 ±3.2	87.2 ±3.1	1.5 ±4.5	√	1.2 ±4.8	√	1.4 ±4.6	√
VOTING	95.1 ±4.1	96.4 ±3.8	95.9 ±4.5	95.6 ±4.9	96.5 ±3.5	0.4 ±4.6	~	-0.9 ±4.4	~	0.1 ±1.8	~
WINE	92.6 ±6.9	92.9 ±6.9	91.4 ±6.3	92.5 ±6.2	92.3 ±6.2	-0.1 ±7.4	~	-0.4 ±8.7	~	-0.6 ±10.0	~
ZOO	95.2 ±7.7	92.2 ±8.0	94.3 ±7.9	94.3 ±6.6	93.5 ±6.9	-1.0 ±8.0	~	2.1 ±10.7	~	1.3 ±10.1	~
NUMERIC XOR-3D	57.7 ±11.1	43.0 ±5.4	89.5 ±11.7	96.1 ±4.3	93.4 ±5.5	38.4 ±12.0	√	53.1 ±7.1	√	50.4 ±7.6	√
NUMERIC XOR-4D	49.8 ±7.0	52.1 ±4.7	62.3 ±11.8	93.2 ±4.4	91.9 ±4.7	43.4 ±7.8	√	41.1 ±6.2	√	39.8 ±5.9	√
MULTIPLEXER-20	61.3 ±7.2	62.1 ±7.5	87.2 ±14.2	95.5 ±8.5	94.5 ±9.5	34.1 ±11.8	√	33.4 ±11.0	√	32.3 ±12.1	√
MULTIPLEX-XOR	58.2 ±12.0	55.4 ±11.5	61.2 ±12.2	76.5 ±11.8	80.1 ±9.6	18.3 ±16.2	√	21.1 ±15.1	√	24.7 ±14.7	√
XOR-5	55.5 ±12.2	54.1 ±12.8	55.8 ±13.2	100.0 ±0.0	100.0 ±0.0	44.5 ±12.2	√	45.9 ±12.8	√	45.9 ±12.8	√
XOR-5 NOISE	54.1 ±12.5	51.7 ±11.9	56.6 ±12.6	74.4 ±11.4	78.2 ±14.0	20.3 ±17.2	√	22.7 ±15.9	√	26.5 ±18.2	√
XOR-10	49.9 ±1.8	49.8 ±1.8	50.3 ±1.6	77.4 ±14.9	79.4 ±16.5	27.5 ±14.7	√	27.7 ±14.8	√	29.6 ±16.7	√

Table 3: The generalization accuracy of the induced trees on various data sets. The numbers represent the average and standard deviation over the individual runs. The last columns list the average differences between the algorithms and their t-test significance, with $\alpha = 0.05$ ($\surd$ indicates a significant advantage and $\times$ a significant disadvantage). The t-test is not applicable for the Monk data sets because only 1 train-test partition was used.

References

Y. Baram, R. El-Yaniv, and K. Luz. Online choice of active learning algorithms. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 19–26, Washington, DC, USA, 2003.

K. Bennett. Global tree optimization: A non-greedy decision tree algorithm. *Computing Science and Statistics*, 26:156–160, 1994.

C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998. URL `http://www.ics.uci.edu/∼mlearn/MLRepository.html`.

A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Occam's Razor. *Information Processing Letters*, 24(6):377–380, 1987.

M. Boddy and T. L. Dean. Deliberation scheduling for problem solving in time constrained environments. *Artificial Intelligence*, 67(2):245–285, 1994.

R. R. Bouckaert. Choosing between two learning algorithms based on calibrated tests. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 51–58, Washington, DC, USA, 2003.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.

W. Buntine and T. Niblett. A further comparison of splitting rules for decision-tree induction. *Machine Learning*, 8(1):75–85, 1992.

O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131–159, 2002.

M. W. Craven. *Extracting Comprehensible Models from Trained Neural Networks*. PhD thesis, Department of Computer Sciences, University of Wisconsin, Madison, 1996.

J. Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

T. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157, 2000.

M. Dong and R. Kothari. Look-ahead based fuzzy decision tree induction. *IEEE-FS*, 9:461–468, June 2001.

J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 194–202, Tahoe City, California, USA, 1995.

S. Esmeir and S. Markovitch. Occam's Razor just got sharper. In *Proceedings of The Twentieth International Joint Conference on Artificial Intelligence*, India, 2007.

F. Esposito, D. Malerba, and G. Semeraro. A comparative analysis of methods for pruning decision trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):476–491, 1997.

U. M. Fayyad and K. B. Irani. What should be minimized in a decision tree? In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 749–754, Boston, Massachusetts, USA, 1990. AAAI Press / The MIT Press.

Y. Freund and L. Mason. The alternating decision tree learning algorithm. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 124–133, Bled, Slovenia, 1999.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag, 2001.

E. Hovitz. *Computation and Action under Bounded Resources*. PhD thesis, Computer Science Department, Stanford University, 1990.

L. Hyafil and R. L. Rivest. Constructing optimal binary decision trees is NP-complete. *Information Processing Letters*, 5(1):15–17, 1976.

G. H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 121–129, 1994.

D.S. Johnson, C.R. Aragon, L.A. McGeoch, and C. Schevon. Optimization by simulated annealing: An experimental evaluation; part ii, graph coloring and number partitioning. *Operations Research*, 39(3):378–406, 1991.

H. Kim and W. Loh. Classification trees with unbiased multiway splits. *Journal of the American Statistical Association*, 96:589–604, 2001.

I. Kononenko, E. Simec, and M. Robnik-Sikonja. Overcoming the myopia of inductive learning algorithms with RELIEFF. *Applied Intelligence*, 7(1):39–55, 1997.

M. Last, A. Kandel, O. Maimon, and E. Eberbach. Anytime algorithm for feature selection. In *Revised Papers from the Second International Conference on Rough Sets and Current Trends in Computing*, pages 532–539. Springer-Verlag, 2001.

M. Lindenbaum, S. Markovitch, and D. Rusakov. Selective sampling for nearest neighbor classifiers. *Machine Learning*, 54(2):125–152, 2004.

D. Lizotte, O. Madani, and R. Greiner. Budgeted learning of naive bayes classifiers. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, Acapulco, Mexico, 2003.

W. Loh and Y. Shih. Split selection methods for classification trees. *Statistica Sinica*, 7:815–840, 1997.

S. Markovitch and D. Rosenstein. Feature generation using general constructor functions. *Machine Learning*, 49:59–98, 2002.

J. Mingers. An empirical comparison of selection measures for decision-tree induction. *Machine Learing*, 3(4):319–342, 1989.

S. Minton, M. D. Johnston, A. B. Philips, and P. Laird. Minimizing conflicts: A heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence*, 58(1-3):161–205, 1992.

T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

O. J. Murphy and R. L. McCraw. Designing storage efficient decision trees. *IEEE Transactions on Computers*, 40(3):315–320, 1991.

P. M. Murphy and M. J. Pazzani. Exploring the Decision Forest: An empirical investigation of Occam's Razor in decision tree induction. *Journal of Artificial Intelligence Research*, 1:257–275, 1994.

S. K. Murthy and S. Salzberg. Lookahead and pathology in decision tree induction. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 1025–1033, Montreal, Canada, 1995.

S. W. Norton. Generating better decision trees. In N. S. Sridharan, editor, *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 800–805, Detroit, Michigan, USA, 1989.

T. Oates and D. Jensen. The effects of training set size on decision tree complexity. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 254–262. Morgan Kaufmann, 1997.

D. Opitz. *An Anytime Approach to Connectionist Theory Refinement: Refining the Topologies of Knowledge-Based Neural Networks*. PhD thesis, Department of Computer Sciences, University of Wisconsin-Madison, 1995.

D. Page and S. Ray. Skewing: An efficient alternative to lookahead for decision tree induction. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, Acapulco, Mexico, 2003.

D. Page and S. Ray. Sequential Skewing: An improved skewing algorithm. In *Proceedings of the Twenty-First International Conference on Machine Learning*, Banff, Canada, 2004.

A. Papagelis and D. Kalles. Breeding decision trees using evolutionary techniques. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 393–400, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, CA, 1993.

J. R. Quinlan and R. M. Cameron-Jones. Oversearching and layered search in empirical learning. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 1019–1024, 1995.

J. R. Quinlan and R. L. Rivest. Inferring decision trees using the minimum description length principle. *Information and Computation*, 80(3):227–248, 1989.

H. Ragavan and L. Rendell. Lookahead feature construction for learning hard concepts. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 252–259, Amherst, MA, USA, 1993.

R. Rao, D. Gordon, and W. Spears. For every generalization action, is there really an equal or opposite reaction? In *Proceedings of Twelfth International Conference on Machine Learning*, pages 471–479, 1995.

S. Ray and D. Page. Generalized Skewing for functions with continuous and nominal attributes. In *Proceedings of the Twenty-Second International Conference on Machine Learning*, Bonn, Germany, 2005.

S. J. Russell and E. Wefald. Principles of metareasoning. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pages 400–411, San Mateo, California, 1989.

S. J. Russell and S. Zilberstein. Optimal composition of real-time systems. *Artificial Intelligence*, 82(1-2):181–213, 1996.

U. K. Sarkar, P. Chakrabarti, S. Ghose, and S. C. DeSarkar. Improving greedy algorithms by lookahead search. *Journal of Algorithms*, 16:1–23, 1994.

A. Schaerf. Tabu search techniques for large high-school timetabling problems. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 363–368, OR, USA, 1996.

C. Schaffer. A conservation law for generalization performance. In *Proceedings of Eleventh International Conference on Machine Learning*, pages 259–265, 1994.

R. Schapire. A brief introduction to boosting. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1401–1406, Stockholm, Sweden, 1999.

Jude W. Shavlik, Raymond J. Mooney, and Geoffrey Towell G. Symbolic and neural learning algorithm: An experimental comparison. *Machine Learning*, 6:111–143, 1991.

P. E. Utgoff. Incremental induction of decision trees. *Machine Learning*, 4(2):161–186, 1989.

P. E. Utgoff, N. C. Berkman, and J. A. Clouse. Decision tree induction based on efficient tree restructuring. *Machine Learning*, 29(1):5–44, 1997.

V. Vapnik. *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

G. I. Webb. Further experimental evidence against the utility of Occam's Razor. *Journal of Artificial Intelligence Research*, 4:397–417, 1996.

I. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, CA, 2005.

Journal of Machine Learning Research 8 (2007) 935-983 Submitted 1/05; Revised 6/06; Published 5/07

Classification in Networked Data: A Toolkit and a Univariate Case Study

Sofus A. Macskassy SOFMAC@FETCH.COM
Fetch Technologies, Inc.
2041 Rosecrans Avenue, Suite 245
El Segundo, CA 90254

Foster Provost FPROVOST@STERN.NYU.EDU
New York University
44 W. 4th Street
New York, NY 10012

Editor: Andrew McCallum

Abstract

This paper[1] is about classifying entities that are interlinked with entities for which the class is known. After surveying prior work, we present NetKit, a modular toolkit for classification in networked data, and a case-study of its application to networked data used in prior machine learning research. NetKit is based on a node-centric framework in which classifiers comprise a local classifier, a relational classifier, and a collective inference procedure. Various existing node-centric relational learning algorithms can be instantiated with appropriate choices for these components, and new combinations of components realize new algorithms. The case study focuses on univariate network classification, for which the only information used is the structure of class linkage in the network (i.e., only links and some class labels). To our knowledge, no work previously has evaluated systematically the power of class-linkage alone for classification in machine learning benchmark data sets. The results demonstrate that very simple network-classification models perform quite well—well enough that they should be used regularly as baseline classifiers for studies of learning with networked data. The simplest method (which performs remarkably well) highlights the close correspondence between several existing methods introduced for different purposes—that is, Gaussian-field classifiers, Hopfield networks, and relational-neighbor classifiers. The case study also shows that there are two sets of techniques that are preferable in different situations, namely when few versus many labels are known initially. We also demonstrate that link selection plays an important role similar to traditional feature selection.

Keywords: relational learning, network learning, collective inference, collective classification, networked data, probabilistic relational models, network analysis, network data

1. Introduction

Networked data contain interconnected entities for which inferences are to be made. For example, web pages are interconnected by hyperlinks, research papers are connected by citations, telephone accounts are linked by calls, possible terrorists are linked by communications. This paper is about

1. Versions of this paper have been available as S.A. Macskassy and Provost, F.J., “Classification in Networked Data: A toolkit and a univariate case study” CeDER Working Paper CeDER-04-08, Stern School of Business, New York University, NY, NY 10012. December 2004. Updated December 2006.

within-network classification: entities for which the class is known are linked to entities for which the class must be estimated. For example, telephone accounts previously determined to be fraudulent may be linked, perhaps indirectly, to those for which no assessment yet has been made.

Such networked data present both complications and opportunities for classification and machine learning. The data are patently not independent and identically distributed, which introduces bias to learning and inference procedures (Jensen and Neville, 2002b). The usual careful separation of data into training and test sets is difficult, and more importantly, thinking in terms of separating training and test sets obscures an important facet of the data: entities with known classifications can serve two roles. They act first as training data and subsequently as background knowledge during inference. Relatedly, within-network inference allows models to use specific node identifiers to aid inference (see Section 3.5.3).

Networked data allow *collective inference*, meaning that various interrelated values can be inferred simultaneously. For example, inference in Markov random fields (MRFs, Dobrushin, 1968; Besag, 1974; Geman and Geman, 1984) uses estimates of a node's neighbors' labels to influence the estimation of the node's label—and vice versa. Within-network inference complicates such procedures by pinning certain values, but also offers opportunities such as the application of network-flow algorithms to inference (see Section 3.5.1). More generally, networked data allow the use of the features of a node's neighbors, although that must be done with care to avoid greatly increasing estimation variance and thereby error (Jensen et al., 2004).

To our knowledge there previously has been no large-scale, systematic experimental study of machine learning methods for within-network classification. A serious obstacle to undertaking such a study is the scarcity of available tools and source code, making it hard to compare various methodologies and algorithms. A systematic study is further hindered by the fact that many relational learning algorithms can be separated into various sub-components; ideally the relative contributions of the sub-components and alternatives should be assessed.

As a main contribution of this paper, we introduce a network learning toolkit (NetKit-SRL) that enables in-depth, component-wise studies of techniques for statistical relational learning and classification with networked data. We abstract prior, published methods into a modular framework on which the toolkit is based.[2]

NetKit is interesting for several reasons. First, various systems from prior work can be realized by choosing particular instantiations for the different components. A common platform allows one to compare and contrast the different systems on equal footing. Perhaps more importantly, the modularity of the toolkit broadens the design space of possible systems beyond those that have appeared in prior work, either by mixing and matching the components of the prior systems, or by introducing new alternatives for components.

In the second half of the paper, we use NetKit to conduct a case study of within-network classification in homogeneous, univariate networks, which are important both practically and scientifically (as we discuss in Section 5). We compare various learning and inference techniques on twelve benchmark data sets from four domains used in prior machine learning research. Beyond illustrating the value of the toolkit, the case study provides systematic evidence that with networked data even univariate classification can be remarkably effective. One implication is that such methods should be used as baselines against which to compare more sophisticated relational learning algorithms (Macskassy and Provost, 2003). One particular very simple and very effective technique

2. NetKit-SRL, or NetKit for short, is written in Java 1.5 and is available as open source from `http://www.research.rutgers.edu/~sofmac/NetKit.html`.

highlights the close correspondence between several types of methods introduced for different purposes: "node-centric" methods, which focus on each node individually; random-field methods; and classic connectionist methods. The case study also illustrates a bias/variance trade-off in networked classification, based on the principle of homophily (the principle that a contact between similar people occurs at a higher rate than among dissimilar people, Blau, 1977; McPherson et al., 2001, page 416; cf., assortativity, Newman, 2003, and relational autocorrelation, Jensen and Neville, 2002b) and suggests network-classification analogues to feature selection and active learning.

To further motivate and to put the rest of the paper into context, we start by reviewing some (published) applications of classification in networked data. Section 3 describes the problem of network learning and classification more formally, introduces the modular framework, and surveys existing work on network classification. Section 4 describes NetKit. Then Section 5 covers the case study, including motivation for studying univariate network inference, the experimental methodology, data used, toolkit components used, and the results and analysis.

2. Applications

The earliest work on classification in networked data arose in scientific applications, with the networks based on regular grids of physical locations. Statistical physics introduced, for example, the Ising model (Ising, 1925) and the Potts model (Potts, 1952), which were used to find minimum energy configurations in physical systems with components exhibiting discrete states, such as magnetic moments in ferromagnetic materials. Network-based techniques then saw application in spatial statistics (Besag, 1974) and in image processing (e.g., Geman and Geman, 1984; Besag, 1986), where the networks were based on grids of pixels.

More recent work has concentrated on networks of arbitrary topology, for example, for the classification of linked documents such as patents (Chakrabarti et al., 1998), scientific research papers (e.g., Taskar et al., 2001; Lu and Getoor, 2003), and web pages (e.g., Neville et al., 2003; Lu and Getoor, 2003). Segal et al. (2003a,b) apply network classification (specifically, relational Markov networks, Taskar et al., 2002) to protein interaction and gene expression data, where protein interactions form a network over which inferences are drawn about pathways, that is, sets of genes that coordinate to achieve a particular task. In computational linguistics network classification is applied to tasks such as the segmentation and labeling of text (e.g., part-of-speech tagging, Lafferty et al., 2001).

Explicit social-network data play an important role in counterterrorism, law enforcement, and fraud detection, because suspicious people may interact with known malicious people. The U.S. Government recently has received attention for its gathering of telephone call-detail records to build a network for counterterrorism analysis (Tumulty, 2006). The simple relational-neighbor technique that performs so well in the case study below (wvRN), combined with a protocol for acquiring network link data, can be applied to communication or surveillance networks for suspicion scoring—ranking candidates by their estimated likelihood of being malicious (Macskassy and Provost, 2005) (cf., Galstyan and Cohen, 2005). In fraud detection, entities to be classified as being fraudulent or legitimate are linked by chains of transactions to those for which classifications are known. For more than a decade "state-of-the-art" fraud detection techniques have included network-based methods such as the "dialed-digit monitor" (Fawcett and Provost, 1997) that examines indirect (two-hop) connections to prior fraudulent accounts in the call network. Cortes et al. (2001) and Hill et al. (2006b) explicitly represent and reason with accounts' local network neighborhoods, for identifying telecommunications fraud. Similarly, networks of relationships between brokers can help in identifying securities fraud (Neville et al., 2005).

For marketing, consumers can be connected into a network based on the products that they buy (or that they rate in a collaborative filtering system), and then network-based techniques can be applied for making product recommendations (Domingos and Richardson, 2001; Huang et al., 2004). If a firm can know actual social-network links between consumers, for example through communications records, statistical, network-based marketing techniques can perform significantly better than traditional targeted marketing based on demographics and prior purchase data (Hill et al., 2006a).

Finally, network classification approaches have seen elegant application to problems that initially do not present themselves as *network* classification. Section 3.5.1 discusses how for "transductive" inference (Vapnik, 1998a), data points can be linked into a network based on any similarity measure. Thus, any transductive classification problem can be treated as a (within-)network classification problem.

3. Network Classification and Learning

Traditionally, machine learning methods have treated entities as being independent, which makes it possible to infer class membership on an entity-by-entity basis. With networked data, the class membership of one entity may have an influence on the class membership of a related entity. Furthermore, entities not directly linked may be related by chains of links, which suggests that it may be beneficial to infer the class memberships of all entities simultaneously. Collective inferencing in relational data (Jensen et al., 2004) makes simultaneous statistical judgments regarding the values of an attribute or attributes for multiple linked entities for which some attribute values are not known.

3.1 Univariate Collective Inferencing

For the univariate case study presented below, the (single) attribute X_i of a vertex v_i, representing the class, can take on some categorical value $c \in \mathcal{X}$—for m classes, $\mathcal{X} = \{c_1, \dots, c_m\}$. We will use c to refer to a non-specified class value.

> Given graph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{X})$ where X_i is the (single) attribute of vertex $v_i \in \mathbf{V}$, and given known values x_i of X_i for some subset of vertices $\mathbf{V}^K$, *univariate collective inferencing* is the process of simultaneously inferring the values x_i of X_i for the remaining vertices, $\mathbf{V}^U = \mathbf{V} - \mathbf{V}^K$, or a probability distribution over those values.

As a shorthand, we will use $\mathbf{x}^K$ to denote the (vector of) class values for $\mathbf{V}^K$, and similarly for $\mathbf{x}^U$. Then, $\mathbf{G}^K = (\mathbf{V}, \mathbf{E}, \mathbf{x}^K)$ denotes everything that is known about the graph (we do not consider the possibility of unknown edges). Edge $e_{ij} \in \mathbf{E}$ represents the edge between vertices v_i and v_j, and w_{ij} represents the edge weight. For this paper we consider only undirected edges (i.e., $w_{ij} = w_{ji}$), if necessary simply ignoring directionality for a particular application.

Rather than estimating the full joint probability distribution $P(\mathbf{x}^U | \mathbf{G}^K)$ explicitly, relational learning often enhances tractability by making a Markov assumption:

$$P(x_i | \mathbf{G}) = P(x_i | \mathcal{N}_i),$$

where $\mathcal{N}_i$ is a set of "neighbors" of vertex v_i such that $P(x_i | \mathcal{N}_i)$ is independent of $\mathbf{G} - \mathcal{N}_i$ (i.e., $P(x_i | \mathcal{N}_i) = P(x_i | \mathbf{G})$). For this paper, we make the ("first-order") assumption that $\mathcal{N}_i$ comprises only the immediate neighbors of v_i in the graph. As one would expect, and as we will see in Section 5.3.5, this assumption can be violated to a greater or lesser degree based on how edges are defined.

Given $\mathcal{N}_i$, a relational model can be used to estimate x_i. Note that $\mathcal{N}_i^U$ $(= \mathcal{N}_i \cap \mathbf{V}^U)$—the set of neighbors of v_i whose values of attribute X are not known—could be non-empty. Therefore, even if the Markov assumption holds, a simple application of the relational model may be insufficient. However, the relational model also may be used to estimate the labels of $\mathcal{N}_i^U$. Further, just as estimates for the labels of $\mathcal{N}_i^U$ influence the estimate for x_i, x_i also influences the estimate of the labels of $v_j \in \mathcal{N}_i^U$ (because edges are undirected, so $v_j \in \mathcal{N}_i \implies v_i \in \mathcal{N}_j$). In order to simultaneously estimate these interdependent values $\mathbf{x}^U$, various collective inference methods can be applied, which we discuss below.

Many of the algorithms developed for within-network classification are heuristic methods without a formal probabilistic semantics (and others are heuristic methods with a formal probabilistic semantics). Nevertheless, let us suppose that at inference time we are presented with a probability distribution structured as a graphical model.[3] In general, there are various inference tasks we might be interested in undertaking (Pearl, 1988). We focus primarily on within-network, univariate classification: the computation of the marginal probability of class membership of a particular node (i.e., the variable represented by the node taking on a particular value), conditioned on knowledge of the class membership of certain other nodes in the network. We also discuss methods for the related problem of computing the maximum a posteriori (MAP) joint labeling for $\mathbf{V}$ or $\mathbf{V}^U$.

For the sort of graphs we expect to encounter in the aforementioned applications, such probabilistic inference is quite difficult. As discussed by Wainwright and Jordan (2003), the naive method of marginalizing by summing over all configurations of the remaining variables is intractable even for graphs of modest size; for binary classification with around 400 unknown nodes, the summation involves more terms than atoms in the visible universe. Inference via belief propagation (Pearl, 1988) is applicable only as a heuristic approximation, because directed versions of many network classification graphs will contain cycles.

An important alternative to heuristic ("loopy") belief propagation is the junction-tree algorithm (Cowell et al., 1999), which provides exact solutions for arbitrary graphs. Unfortunately, the computational complexity of the junction-tree algorithm is exponential in the "treewidth" of the junction tree formed by the graph (Wainwright and Jordan, 2003). Since the treewidth is one less than the size of the largest clique, and the junction tree is formed by triangulating the original graph, the complexity is likely to be prohibitive for graphs such as social networks, which can have dense local connectivity and long cycles.

3.2 A Node-centric Network Learning Framework and Historical Background: Local, Relational, and Collective Inference

A large set of approaches to the problem of network classification can be viewed as "node centric," in the sense that they focus on a single node at a time. For a couple reasons, which we elaborate presently, it is useful to divide such systems into three components. One component, the *relational classifier*, addresses the question: given a node and the node's neighborhood, how should a classification or a class-probability estimate be produced? For example, the relational classifier might

3. For this paper, we assume that the structure of the network resulting from the chosen links corresponds at least partially to the structure of the network of probabilistic dependencies. This of course will be more or less true based on the choice of links, as we will see in Section 5.3.5.

1. **Non-relational ("local") model.** This component consists of a (learned) model, which uses only local information—namely information about (attributes of) the entities whose target variable is to be estimated. The local models can be used to generate priors that comprise the initial state for the relational learning and collective inference components. They also can be used as one source of evidence during collective inference. These models typically are produced by traditional machine learning methods.
2. **Relational model.** In contrast to the non-relational component, the relational model makes use of the relations in the network as well as the values of attributes of related entities, possibly through long chains of relations. Relational models also may use local attributes of the entities.
3. **Collective inferencing.** The collective inferencing component determines how the unknown values are estimated together, possibly influencing each other, as described above.

Table 1: The three main components making up a (node-centric) network learning system.

combine local features and the labels of neighbors using a naive Bayes model (Chakrabarti et al., 1998) or a logistic regression (Lu and Getoor, 2003). A second component addresses the problem of *collective inference*: what should we do when a classification depends on a neighbor's classification, and vice versa? Finally, most such methods require initial ("prior") estimates of the values for $P(\mathbf{x}^U|\mathbf{G}^K)$. The priors could be Bayesian subjective priors (Savage, 1954), or they could be estimated from data. A common estimation method is to employ a non-relational learner, using available "local" attributes of v_i to estimate x_i (e.g., as done by Besag, 1986). We propose a general "node-centric" network classification framework consisting of these three main components, listed in Table 1.

Viewing network classification approaches through this decomposition is useful for two main reasons. First, it provides a way of describing certain approaches that highlights the similarities and differences among them. Secondly, it expands the small set of existing methods to a design space of methods, since components can be mixed and matched in new ways. In fact, some novel combination may well perform better than those previously proposed; there has been little systematic experimentation along these lines. Local and relational classifiers can be drawn from the vast space of classifiers introduced over the decades in machine learning, statistics, pattern recognition, etc., and treated in great detail elsewhere. Collective inference has received much less attention in all these fields, and therefore warrants additional introduction.

Collective inference has its roots mainly in pattern recognition and statistical physics. Markov random fields have been used extensively for univariate network classification for vision and image restoration. Introductions to MRFs fill textbooks (Winkler, 2003); for our purposes, it is important to point out that they are the basis both directly and indirectly for many network classification approaches. MRFs are used to estimate a joint probability distribution over the free variables of a set of nodes under the first-order Markov assumption that $P(x_i|\mathbf{G}/v_i) = P(x_i|\mathcal{N}_i)$, where x_i is the (estimated) label of vertex v_i, $\mathbf{G}/v_i$ means all nodes in $\mathbf{G}$ except v_i, and $\mathcal{N}_i$ is a neighborhood function returning the neighbors of v_i. In a typical image application, nodes in the network are pixels and the labels are image properties such as whether a pixel is part of a vertical or horizontal border.

Because of the obvious interdependencies among the nodes in an MRF, computing the joint probability of assignments of labels to the nodes ("configurations") requires collective inference. Gibbs sampling (Geman and Geman, 1984) was developed for this purpose for restoring degraded images. Geman and Geman enforce that the Gibbs sampler settles to a final state by using simulated annealing where the temperature is dropped slowly until nodes no longer change state. Gibbs sampling is discussed in more detail below.

Two problems with Gibbs sampling (Besag, 1986) are particularly relevant for machine learning applications of network classification. First, prior to Besag's paper Gibbs sampling typically was used in vision not to compute the final marginal posteriors, as required by many "scoring" applications where the goal is to rank individuals, but rather to get final MAP classifications. Second, Gibbs sampling can be very time consuming, especially for large networks (not to mention the problems detecting convergence in the first place). With his Iterated Conditional Modes (ICM) algorithm, Besag introduced the notion of *iterative classification* for scene reconstruction. In brief, iterative classification repeatedly classifies labels for $v_i \in \mathbf{V}^U$, based on the "current" state of the graph, until no vertices change their label. ICM is presented as being efficient and particularly well suited to maximum marginal classification by node (pixel), as opposed to maximum joint classification over all the nodes (the scene).

Two other, closely related, collective inference techniques are (loopy) belief propagation (Pearl, 1988) and relaxation labeling (Rosenfeld et al., 1976; Hummel and Zucker, 1983). Loopy belief propagation was introduced above. Relaxation labeling originally was proposed as a class of parallel iterative numerical procedures that use contextual constraints to reduce ambiguities in image analysis; an instance of relaxation labeling is described in detail below. Both methods use the estimated class distributions directly, rather than the hard labelings used by iterative classification. Therefore, one requirement for applying these methods is that the relational classifier, when estimating x_i, must be able to use the estimated class distributions of $v_j \in N_i^U$.

Graph-cut techniques recently have been used in vision research as an alternative to using Gibbs sampling (Boykov et al., 2001). In essence, these are collective inference procedures, and are the basis of a collection of modern machine learning techniques. However, they do not quite fit in the node-centric framework, so we treat them separately below.

3.3 Node-centric Network Classification Approaches

The node-centric framework allows us to describe several prior systems by how they solve the problems of local classification, relational classification, and collective inference. The components of these systems are the basis for composing methods in NetKit.

For classifying web-pages based on the text and (possibly inferred) class labels of neighboring pages, Chakrabarti et al. (1998) combined naive Bayes local and relational classifiers with relaxation labeling for collective inference. In their experiments, performing network classification using the web-pages' link structure substantially improved classification as compared to using only the local (text) information. Specifically, considering the text of neighboring pages generally hurt performance, whereas using only the (inferred) class labels improved performance.

The iterated conditional modes procedure (ICM, Besag, 1986) is a node-centric approach where the local and relational classifiers are domain-dependent probabilistic models (based on local attributes and a MRF), and iterative classification is used for collective inference. Iterative classification has been used for collective inference elsewhere as well, for example Neville and Jensen (2000) use it in combination with naive Bayes for local and relational classification (with a simulated annealing procedure to settle on the final labeling).

We will look in more detail at the procedure known as "link-based classification" (Lu and Getoor, 2003), also introduced for the classification of linked documents (web pages and published manuscripts with an accompanying citation graph). Similarly to the work of Chakrabarti et al. (1998), linked-based classification uses the (local) text of the document as well as neighbor labels. More specifically, the relational classifier is a logistic regression model applied to a vector of aggregations of properties of the sets of neighbor labels linked with different types of links (in-, out-, co-links). Various aggregates could be used and are examined by Lu and Getoor (2003), such as the mode (the value of the most often occurring neighbor class), a binary vector with a value of 1 at cell i if there was a neighbor whose class label was c_i, and a count vector where cell i contained the number of neighbors belonging to class c_i. In their experiments, the count model performed best. They used logistic regression on the local (text) attributes of the instances to initialize the priors for each vertex in their graph and then applied the link-based classifiers as their relational model.

The simplest network classification technique we will consider was introduced to highlight the remarkable amount of "power" for classification present just in the structure of the network, a notion that we will investigate in depth in the case study below. The weighted-vote relational neighbor (wvRN) procedure (Macskassy and Provost, 2003) performs relational classification via a weighted average of the (potentially estimated) class membership scores ("probabilities") of the node's neighbors. Collective inference is performed via a relaxation labeling method similar to that used by Chakrabarti et al. (1998). If local attributes such as text are ignored, the node priors can be instantiated with the unconditional marginal class distribution estimated from the training data.

Since wvRN performs so well in the case study below, it is noteworthy to point out its close relationship to Hopfield networks (Hopfield, 1982) and Boltzmann machines (Hinton and Sejnowski, 1986). A Hopfield network is a graph of homogeneous nodes and undirected edges, where each node is a binary threshold unit. Hopfield networks were designed to recover previously seen graph configurations from a partially observed configuration, by repeatedly estimating the states of nodes one at a time. The state of a node is determined by whether or not its input exceeds its threshold, where the input is the weighted sum of the states of its immediate neighbors. wvRN differs in that it retains uncertainty at the nodes rather than assigning each a binary state (also allowing multi-class networks). *Learning* in Hopfield networks consists of learning the weights of edges and the thresholds of nodes, given one or more input graphs. Given a partially observed graph state and repeatedly applying, node-by-node, the node-activation equation will provably converge to a stable graph state—the low-energy state of the graph. If the partial input state is "close" to one of the training states, the Hopfield network will converge to that state.

A Boltzmann machine, like a Hopfield network, is a network of units with an "energy" defined for the network (Hinton and Sejnowski, 1986). Unlike Hopfield networks, Boltzmann machine nodes are stochastic and the machines use simulated annealing to find a stable state. Boltzmann machines also often have both visible and hidden nodes. The visible nodes' states can be observed, whereas the states of the hidden nodes cannot—as with hidden Markov models.

3.4 Modeling Homophily for Classification

The case study below demonstrates the remarkable power of a simple assumption: linked entities have a propensity to belong to the same class. This autocorrelation in the class variable of related entities is one form of homophily (the principle that a contact between similar people occurs at a higher rate than among dissimilar people), which is ubiquitous in observations and theories of social networks (Blau, 1977; McPherson et al., 2001). The relational neighbor classifier, which performs well in the case study below, was introduced as a very simple classifier based solely on homophily, that might provide a baseline to other relational classification techniques (Macskassy and Provost, 2003). As we discuss in Section 3.5.4 some more complex relational classification techniques can deal well with (arbitrary) homophily, and others cannot.

Homophily was one of the first characteristics noted by early social network researchers (Almack, 1922; Bott, 1928; Richardson, 1940; Loomis, 1946; Lazarsfeld and Merton, 1954), and holds for a wide variety of different relationships (McPherson et al., 2001). It seems reasonable to conjecture that homophily may also be present in other sorts of networks, especially networks of artifacts created by people. (Recently *assortativity*, a link-centric notion of homophily, has become the focus of mathematical studies of network structure, Newman, 2003).

Shared membership in groups such as communities with shared interests is an important reason for similarity among interconnected nodes (Neville and Jensen, 2005). Inference can be more effective if these groups are modeled explicitly, such as by using latent group models (LGMs, Neville and Jensen, 2005) to specify joint models of attributes, links, and *groups*. LGMs are especially promising for within-network classification, since the existence of known classes will facilitate the identification of (hidden) group membership, which in turn may aid in the estimation of $\mathbf{x}^U$.

3.5 Other Methods for Network Classification

Before describing the node-centric network classification toolkit, for completeness we first will discuss three other types of methods that are suited to univariate, within-network classification. Graph-based methods (Sections 3.5.1 and 3.5.2) that are used for semi-supervised learning, could apply as well to within-network classification. Within-network classification also offers the opportunity to take advantage of node identifiers, discussed in Section 3.5.3. Finally, although there are important reasons to study univariate network classification (see below), recently the field has seen a flurry of development of multivariate methods applicable to classification in networked data (Section 3.5.4).

3.5.1 GRAPH-CUT METHODS AND THEIR RELATIONSHIP TO TRANSDUCTIVE INFERENCE

As mentioned above, one complication to within-network classification is that the to-be-classified nodes and the nodes for which the labels are known are intermixed in the same network. Most prior work on network learning and classification assumes that the classes of all the nodes in the network need to be estimated (perhaps having learned something from a separate, related network). Pinning the values of certain nodes intuitively should be advantageous, since it gives to the classification procedure clear points of reference.

This complication is addressed directly by several lines of recent work (Blum and Chawla, 2001; Joachims, 2003; Zhu et al., 2003; Blum et al., 2004). The setting is not initially one of network classification, but rather, semi-supervised learning in a transductive setting (Vapnik, 1998b). Nevertheless, the methods introduced may have direct application to certain instances of univariate network classification. Specifically, they consider data sets where labels are given for a subset of cases, and classifications are desired for a subset of the rest. To form an "induced" weighted network, edges are added between data points based on similarity between cases.

Finding the minimum energy configuration of a MRF, the partition of the nodes that maximizes self-consistency under the constraint that the configuration be consistent with the known labels, is equivalent to finding a minimum cut of the graph (Greig et al., 1989). Following this idea and subsequent work connecting classification to the problem of computing minimum cuts (Kleinberg and Tardos, 1999), Blum and Chawla (2001) investigate how to define weighted edges for a transductive classification problem such that polynomial-time mincut algorithms give optimal solutions to objective functions of interest. For example, they show elegantly how forms of leave-one-out-cross-validation error (on the predicted labels) can be minimized for various nearest-neighbor algorithms, including a weighted-voting algorithm. This procedure corresponds to optimizing the consistency of the predictions in particular ways—as Blum and Chawla put it, optimizing the "happiness" of the classification algorithm.

Of course, optimizing the consistency of the labeling may not be ideal. For example in the case of a highly unbalanced class frequency it is necessary to preprocess the graph to avoid degenerate cuts, for example those cutting off the one positive example (Joachims, 2003). This seeming pathology stems from the basic objective: the minimum of the sum of cut-through edge weights depends directly on the sizes of the cut sets; normalizing for the cut size leads to ratiocut optimization (Dhillon, 2001) constrained by the known labels (Joachims, 2003).

The mincut partition corresponds to the most probable joint labeling of the graph (taking an MRF perspective), whereas as discussed earlier we often would like a per-node (marginal) class-probability estimation (Blum et al., 2004). Unfortunately, in the case we are considering—when some node labels are known in a general graph—there is no known efficient algorithm for determining these estimates. There are several other drawbacks (Blum et al., 2004), including that there may be many minimum cuts for a graph (from which mincut algorithms choose rather arbitrarily), and that the mincut approach does not yield a measure of confidence on the classifications. Blum et al. address these drawbacks by repeatedly adding artificial noise to the edge weights in the induced graph. They then can compute fractional labels for each node corresponding to the frequency of labeling by the various mincut instances. As mentioned above, this method (and the one discussed next) was intended to be applied to an induced graph, which can be designed specifically for the application. Mincut approaches are appropriate for graphs that have at least some small, balanced cuts (whether or not these correspond to the labeled data) (Blum et al., 2004). It is not clear whether methods like this that discard highly unbalanced cuts will be effective for network classification problems such as fraud detection in transaction networks, with extremely unbalanced class distributions.

3.5.2 THE GAUSSIAN-FIELD CLASSIFIER

In the experiments of Blum et al. (2004), their randomized mincut method empirically does not perform as well as a method introduced by Zhu et al. (2003). Therefore, we will revisit this latter method in an experimental comparison following the main case study. Zhu et al. treat the induced network as a Gaussian field (Besag, 1975) (a random field with soft node labels) constrained such that the labeled nodes maintain their values. The value of the energy function is the weighted average of the function's values at the neighboring points. Zhu et al. show that this function is a harmonic function and that the solution over the complete graph can be computed using a few matrix operations. The result is a classifier essentially identical to the wvRN classifier (Macskassy and Provost, 2003) discussed above (paired with relaxation labeling), except with a principled semantics and exact inference.[4] The energy function then can be normalized based on desired class posteriors ("class mass normalization"). Zhu et al. also discuss various physical interpretations of this procedure, including random walks, electric networks, and spectral graph theory, that can be intriguing in the context of particular applications. For example, applying the random walk interpretation to a telecommunications network including legitimate and fraudulent accounts: consider starting at an account of interest and walking randomly through the call graph based on the link weights; the node score is the probability that the walk hits a known fraudulent account before hitting a known legitimate account.

3.5.3 USING NODE IDENTIFIERS

As mentioned in the introduction, another unique aspect of within-network classification is that *node identifiers*, unique symbols for individual nodes, can be used in learning and inference. For example, for suspicion scoring in social networks, the fact that someone met with a particular individual may be informative (e.g., having had repeated meetings with a known terrorist leader). Very little work has incorporated identifiers, because of the obvious difficulty of modeling with very high cardinality categorical attributes. Identifiers (telephone numbers) have been used for fraud detection (Fawcett and Provost, 1997; Cortes et al., 2001; Hill et al., 2006b), but to our knowledge, Perlich and Provost (2006) provide the only comprehensive treatment of the use of identifiers for relational learning.

3.5.4 BEYOND UNIVARIATE CLASSIFICATION

Besides the methods already discussed (e.g., Besag, 1986; Lu and Getoor, 2003; Chakrabarti et al., 1998), several other methods go beyond the homogeneous, univariate case on which this paper focuses. Conditional random fields (CRFs, Lafferty et al., 2001) are random fields where the probability of a node's label is conditioned not only on the labels of neighbors (as in MRFs), but also on all the observed attribute data.

Relational Bayesian networks (RBNs, a.k.a. probabilistic relational models, Koller and Pfeffer, 1998; Friedman et al., 1999; Taskar et al., 2001) extend Bayesian networks (BNs, Pearl, 1988) by taking advantage of the fact that a variable used in one instantiation of a BN may refer to the exact same variable in another BN. For example, consider that the grade of a student depends to some extent upon his professor; this professor is the same for all students in the class. Therefore, rather than building one BN and using it in isolation for each entity, RBNs directly link shared variables, thereby generating one big network of connected entities for which collective inferencing can be performed.

Unfortunately, because the BN representation must be acyclic, RBNs cannot model arbitrary relational autocorrelation, such as the homophily that plays a large role in the case study below. However, undirected relational graphical models can model relational autocorrelation. Relational dependency networks (RDNs, Neville and Jensen, 2003, 2004, 2007), extend dependency networks (Heckerman et al., 2000) in much the same way that RBNs extend Bayesian networks. RDNs learn the dependency structure by learning a conditional model individually for each variable of interest, conditioning on the other variables, including variables of other nodes in the network. Cyclic de-

4. Experiments show these two procedures to yield almost identical generalization performance, albeit the matrix-based procedure of Zhu et al. is much slower than the iterative wvRN.

```
Input: $\mathbf{G}^K, \mathbf{V}^U, \text{RC}_{\text{type}}, \text{LC}_{\text{type}}, \text{CI}_{\text{type}}$
Induce a local classification model, LC, of type $\text{LC}_{\text{type}}$, using $\mathbf{G}^K$
Induce a relational classification model, RC, of type $\text{RC}_{\text{type}}$, using $\mathbf{G}^K$
Estimate $x_i \in \mathbf{V}^U$ using LC.
Apply collective inferencing of type $\text{CI}_{\text{type}}$, using RC as the model.
Output: Final estimates for $x_i \in \mathbf{V}^U$
```

Table 2: High-level pseudo-code for the core routine of the Network Learning Toolkit.

pendencies such as homophily are modeled when the conditional modeling for a particular variable includes instantiations of the same variable from linked nodes. Relational extensions to Markov networks (Pearl, 1988) also can model arbitrary autocorrelation dependencies. In relational Markov networks (RMNs, Taskar et al., 2002) the clique potential functions are based on functional templates, each of which is a (learned, class-conditional) probability function based on a user-specified set of relations. In associative Markov networks (AMNs, Taskar et al., 2004) autocorrelation of labels is explicitly modeled by extending the *generalized Potts model* (Potts, 1952) (specifically, to allow different labels to have different penalties). Of particular interest, exact inference can be performed in AMNs (formulated as a quadratic programming problem for binary classification, which can be relaxed for multi-class problems).

These methods for network classification use only a few of the many relational learning techniques. There are many more, for example from the rich literature of inductive logic programming (ILP, such as, De Raedt et al., 2001; Dzeroski and Lavrac, 2001; Kramer et al., 2001; Flach and Lachiche, 2004; Richardson and Domingos, 2006), or based on using relational database joins to generate relational features (e.g., Perlich and Provost, 2003; Popescul and Ungar, 2003; Perlich and Provost, 2006).

4. Network Learning Toolkit (NetKit-SRL)

NetKit is designed to accommodate the interchange of components and the introduction of new components. As outlined in Section 3.2, the node-centric learning framework comprises three main components: the relational classifier, the local classifier and collective inference. Any local classifier can be paired with any relational classifier, which can then be combined with any collective inference method. NetKit's core routine is simple and is outlined in Table 2; it consists of these five general modules:

1. **Input:** This module reads data into a memory-resident graph $\mathbf{G}$.

2. **Local classifier inducer:** Given as training data $\mathbf{V}^K$, this module returns a model $\mathcal{M}_L$ that will estimate x_i using only attributes of a node $v_i \in \mathbf{V}^U$. Ideally, $\mathcal{M}_L$ will estimate a probability distribution over the possible values for x_i.

3. **Relational classifier inducer:** Given $\mathbf{G}^K$, this module returns a model $\mathcal{M}_R$ that will estimate x_i using v_i and $\mathcal{N}_i$. Ideally, $\mathcal{M}_R$ will estimate a probability distribution over the possible values for x_i.

4. **Collective Inferencing:** Given a graph $\mathbf{G}$ possibly with some x_i known, a set of prior estimates for $\mathbf{x}^U$, and a relational model $\mathcal{M}_R$, this module applies collective inferencing to estimate $\mathbf{x}^U$.

5. **Weka Wrapper:** This module is a wrapper for Weka[5] (Witten and Frank, 2000) and can convert the graph representation of v_i into an entity that can either be learned from or be used to estimate x_i. NetKit can use a Weka classifier either as a local classifier or as a relational classifier (by using various aggregation methods to summarize the values of attributes in $\mathcal{N}_i$).

The current version of NetKit-SRL, while able to read in heterogeneous graphs, only supports classification in graphs consisting of a single type of node. Algorithms based on expectation maximization are possible to implement through the NetKit collective inference module, by having the collective inference module repeatedly apply a relational classifier to learn a new relational model and then apply the new relational model to $\mathbf{G}$ (rather than repeatedly apply the same learned model at every iteration).

The rest of this section describes the particular relational classifiers and collective inference methods implemented in NetKit for the case study. First, we describe the four (univariate[6]) relational classifiers. Then, we describe the three collective inference methods.

4.1 Relational Classifiers

All four relational classifiers take advantage of the first-order Markov assumption on the network: only a node's local neighborhood is necessary for classification. The univariate case renders this assumption particularly restrictive: only the class labels of the local neighbors are necessary. The local network is defined by the user, analogous to the user's definition of the feature set for propositional learning. Entities whose class labels are not known are either ignored or are assigned a prior probability, depending upon the choice of local classifier.

4.1.1 WEIGHTED-VOTE RELATIONAL NEIGHBOR CLASSIFIER (WVRN)

The case study's simplest classifier (Macskassy and Provost, 2003)[7] estimates class-membership probabilities by assuming the existence of homophily (see Section 3.4).

Definition. Given $v_i \in \mathbf{V}^U$, the weighted-vote relational-neighbor classifier (wvRN) estimates $P(x_i|\mathcal{N}_i)$ as the (weighted) mean of the class-membership probabilities of the entities in $\mathcal{N}_i$:

$$P(x_i = c|\mathcal{N}_i) = \frac{1}{Z} \sum_{v_j \in \mathcal{N}_i} w_{i,j} \cdot P(x_j = c|\mathcal{N}_j),$$

where Z is the usual normalizer. This can be viewed simply as an inference procedure, or as a probability model. In the latter case, it corresponds exactly to the Gaussian-field model discussed below in Section 3.5.2. How to conduct the inference/estimate the model falls on the collective inference procedure.

5. We use version 3.4.2. Weka is available at `http://www.cs.waikato.ac.nz/~ml/weka/`.
6. The open-source NetKit release contains multivariate versions of these classifiers.
7. Previously called the probabilistic Relational Neighbor classifier (pRN).

4.1.2 CLASS-DISTRIBUTION RELATIONAL NEIGHBOR CLASSIFIER (CDRN)

The simple wvRN assumes that neighboring class labels were likely to be the same. *Learning* a model of the distribution of neighbor class labels is more flexible, and may lead to better discrimination. Following Perlich and Provost (2003, 2006), and in the spirit of Rocchio's method (Rocchio, 1971), we define node v_i's *class vector* $\text{CV}(v_i)$ to be the vector of summed linkage weights to the various (known) classes, and class c's *reference vector* $\text{RV}(c)$ to be the average of the class vectors for nodes known to be of class c. Specifically:

$$\text{CV}(v_i)_k = \sum_{v_j \in \mathcal{N}_i, x_j = c_k} w_{i,j}, \tag{1}$$

where $\text{CV}(v_i)_k$ represents the kth position in the class vector and $c_k \in \mathcal{X}$ is the kth class. Based on these class vectors, the reference vectors can then be defined as the normalized vector sum:

$$\text{RV}(c) = \frac{1}{|\mathbf{V}_c^K|} \sum_{v_i \in \mathbf{V}_c^K} \text{CV}(v_i), \tag{2}$$

where $\mathbf{V}_c^K = \{v_i | v_i \in \mathbf{V}^K, x_i = c\}$.

For the case study, during training, neighbors in $\mathbf{V}^U$ are ignored. For prediction, estimated class membership probabilities are used for neighbors in $\mathbf{V}^U$, and Equation 1 becomes:

$$\text{CV}(v_i)_k = \sum_{v_j \in \mathcal{N}_i} w_{i,j} \cdot P(x_j = c_k | \mathcal{N}_j). \tag{3}$$

Definition. Given $v_i \in \mathbf{V}^U$, the *class-distribution relational-neighbor classifier (cdRN)* estimates the probability of class membership, $P(x_i = c | \mathcal{N}_i)$, by the normalized vector similarity between v_i's class vector and class c's reference vector:

$$P(x_i = c | \mathcal{N}_i) = \text{sim}(\text{CV}(v_i), \text{RV}(c)),$$

where $\text{sim}(a, b)$ is any vector similarity function (L_1, L_2, cosine, etc.), normalized to lie in the range $[0, 1]$. For the results presented below, we use cosine similarity.

As with wvRN, Equation 3 is a recursive definition, and therefore the value of $P(x_j = c | \mathcal{N}_j)$ is approximated by the "current" estimate as defined by the selected collective inference technique.

4.1.3 NETWORK-ONLY BAYES CLASSIFIER (NBC)

NetKit's network-only Bayes classifier (nBC) is based on the algorithm described by Chakrabarti et al. (1998). To start, assume there is a single node v_i in $\mathbf{V}^U$. The nBC uses multinomial naive Bayesian classification based on the classes of v_i's neighbors.

$$P(x_i = c | \mathcal{N}_i) = \frac{P(\mathcal{N}_i | c) \cdot P(c)}{P(\mathcal{N}_i)},$$

where

$$P(\mathcal{N}_i | c) = \frac{1}{Z} \prod_{v_j \in \mathcal{N}_i} P(x_j = \tilde{x}_j | x_i = c)^{w_{i,j}}$$

where Z is a normalizing constant and $\tilde{x}_j$ is the class observed at node v_j. As usual, because $P(\mathcal{N}_i)$ is the same for all classes, normalization across the classes allows us to avoid explicitly computing it.

We call nBC "network-only" to emphasize that in the application to the univariate case study below, we do not use local attributes of a node. As discussed above, Chakrabarti et al. initialize nodes' priors based on a naive Bayes model over the local document text and add a text-based term to the node probability formula.[8] In the univariate setting, local text is not available. We therefore use the same scheme as for the other relational classifiers: initialize unknown labels as decided by the local classifier being used (in our study: either the class prior or `'null'`, depending on the collective inference component, as described below). If a neighbor's label is `'null'`, then it is ignored for classification. Also, Chakrabarti et al. differentiated between incoming and outgoing links, whereas we do not. Finally, Chakrabarti et al. do not mention how or whether they account for possible zeros in the estimations of the marginal conditional probabilities; we apply traditional Laplace smoothing where $m = |\mathcal{X}|$, the number of classes.

The foregoing assumes all neighbor labels are known. When the values of some neighbors are unknown, but estimations are available, we follow Chakrabarti et al. (1998) and perform a Bayesian combination based on (estimated) configuration priors and the entity's known neighbors. Chakrabarti et al. (1998) describe this procedure in detail. For our case study, such an estimation is necessary only when using relaxation labeling (described below).

4.1.4 NETWORK-ONLY LINK-BASED CLASSIFICATION (nLB)

The final relational classifier used in the case study is a network-only derivative of the link-based classifier (Lu and Getoor, 2003). The network-only Link-Based classifier (nLB) creates a feature vector for a node by aggregating the labels of neighboring nodes, and then uses logistic regression to build a discriminative model based on these feature vectors. This learned model is then applied to estimate $P(x_i = c|\mathcal{N}_i)$. As with the nBC, the difference between the "network-only" link-based classifier and Lu and Getoor's version is that for the univariate case study we do not consider local attributes (e.g., text).

As described above, Lu and Getoor (2003) considered various aggregation methods: existence (binary), the mode, and value counts. The last aggregation method, the count model, is equivalent to the class vector $\mathrm{CV}(v_i)$ defined in Equation 3. This was the best performing method in the study by Lu and Getoor, and is the method on which we base nLB. The logistic regression classifier used by nLB is the multiclass implementation from Weka version 3.4.2.

We made one minor modification to the original link-based classifier. Perlich (2003) argues that in different situations it may be preferable to use either vectors based on raw counts (as given above) or vectors based on normalized counts. We did preliminary runs using both. The normalized vectors generally performed better, and so we use them for the case study.

4.2 Collective Inference Methods

This section describes three collective inferencing methods implemented in NetKit and used in the case study. As described above, given (i) a network initialized by the local model, and (ii) a relational model, a collective inference method infers a set of class labels for $\mathbf{x}^U$. Depending

8. The original classifier was defined as: $P(x_i = c|\mathcal{N}_i) = P(\mathcal{N}_i|c) \cdot P(\tau_i|v_i) \cdot P(c)$, with τ_i being the text of the document-entity represented by vertex v_i.

```
1. Initialize $v_i \in \mathbf{V}^U$ using the local classifier model, $\mathcal{M}_L$. Specifically, for $v_i \in \mathbf{V}^U$:
    (a) $\widehat{\mathbf{c}}_i \leftarrow \mathcal{M}_L(v_i)$, where $\widehat{\mathbf{c}}_i$ is a vector of probabilities representing $\mathcal{M}_L$'s estimates
        of $P(x_i|\mathcal{N}_i)$. We use $\widehat{\mathbf{c}}_i(k)$ to mean the $k^{\text{th}}$ value in $\widehat{\mathbf{c}}_i$, which represents
        $P(x_i = c_k|\mathcal{N}_i)$.

        For the case study, the local classifier model returns the unconditional
        marginal class distribution estimated from $\mathbf{x}^K$.
    (b) Sample a value $c_s$ from $\widehat{\mathbf{c}}_i$, such that $P(c_s = c_k|\widehat{\mathbf{c}}_i) = \widehat{\mathbf{c}}_i(k)$.
    (c) Set $x_i \leftarrow c_s$.
2. Generate a random ordering, $O$, of vertices in $\mathbf{V}^U$.
3. For elements $v_i \in O$ in order:
    (a) Apply the relational classifier model: $\widehat{\mathbf{c}}_i \leftarrow \mathcal{M}_R(v_i)$.
    (b) Sample a value $c_s$ from $\widehat{\mathbf{c}}_i$, such that $P(c_s = c_k|\widehat{\mathbf{c}}_i) = \widehat{\mathbf{c}}_i(k)$.
    (c) Set $x_i \leftarrow c_s$.
        Note that when $\mathcal{M}_R$ is applied to estimate $\widehat{\mathbf{c}}_\mathbf{i}$ it uses the "new" labelings
        from elements $1, \ldots, (i-1)$, while using the "current" labelings for elements
        $(i+1), \ldots, n$.
4. Repeat prior step 200 times without keeping any statistics. This is known as the
   burnin period.
5. Repeat again for 2000 iterations, counting the number of times each $X_i$ is assigned
   a particular value $c \in \mathcal{X}$. Normalizing these counts forms the final class probability
   estimates.
```

Table 3: Pseudo-code for Gibbs sampling.

on the application, the goal ideally would be to infer the class labels with either the maximum joint probability or the maximum marginal probability for each node. Alternatively, if estimates of entities' class-membership probabilities are needed, the collective inference method estimates the marginal probability distribution $P(X_i = c|\mathbf{G}^K, \Lambda)$ for each $X_i \in \mathbf{x}^U$ and $c \in \mathcal{X}$, where Λ stands for the priors returned by the local classifier.

4.2.1 GIBBS SAMPLING (GS)

Gibbs sampling (Geman and Geman, 1984) is commonly used for collective inferencing with relational learning systems. The algorithm is straightforward and is shown in Table 3.[9] The use of 200 and 2000 for the burnin period and number of iterations are commonly used values.[10] Ideally, we would iterate until the estimations converge. Although there are convergence tests for the Gibbs sampler, they are neither robust nor well understood (cf., Gilks et al., 1995), so we simply use a fixed number of iterations.

9. This instance of Gibbs sampling uses a single random ordering ("chain"), as this is what we used in the case study. In the case study, averaging over 10 chains (the default in NetKit) had no effect on the final accuracies.

10. As it turns out, in our case study Gibbs sampling invariably reached a seemingly final plateau in fewer than 1000 iterations, and often in fewer than 500.

```
1. For $v_i \in \mathbf{V}^U$, initialize the prior: $\widehat{\mathbf{c}}_i^{(0)} \leftarrow \mathcal{M}_L(v_i)$, where $\widehat{\mathbf{c}}_i$ is defined as in Table 3.

   For the case study, the local classifier model returns the unconditional marginal class
   distribution estimated from $\mathbf{x}^K$.
2. For elements $v_i \in \mathbf{V}^U$:
   (a) Estimate $x_i$ by applying the relational model:

           $\widehat{\mathbf{c}}_i^{(t+1)} \leftarrow \mathcal{M}_R(v_i^{(t)}),$     (4)

       where $\mathcal{M}_R(v_i^{(t)})$ denotes using the estimates $\widehat{\mathbf{c}}_j^{(t)}$ for $v_j \in \mathcal{N}_i$, and $t$ is the
       iteration count. This has the effect that all predictions are done pseudo-
       simultaneously based on the state of the graph after iteration $t$.
3. Repeat for $T$ iterations, where $T = 99$ for the case study. $\widehat{\mathbf{c}}^{(T)}$ will comprise the final
   class probability estimations.
```

Table 4: Pseudo-code for relaxation labeling.

Notably, because all nodes are assigned a class at every iteration, when Gibbs sampling is used the relational models will always see a fully labeled/classified neighborhood, making prediction straightforward. For example, nBC does not need to compute its Bayesian combination (see Section 4.1.3).

4.2.2 RELAXATION LABELING (RL)

The second collective inferencing method used in the study is relaxation labeling, based on the method of Chakrabarti et al. (1998). Rather than treat $\mathbf{G}$ as being in a specific labeling "state" at every point (as Gibbs sampling does), relaxation labeling retains the uncertainty, keeping track of the current probability estimations for $\mathbf{x}^U$. The relational model must be able to use these estimations. Further, rather than estimating one node at a time and updating the graph right away, relaxation labeling "freezes" the current estimations so that at step $t+1$ all vertices will be updated based on the estimations from step t. The algorithm is shown in Table 4.

Preliminary runs showed that relaxation labeling sometimes does not converge, but rather ends up oscillating between two or more graph states.[11] NetKit performs simulated annealing—on each subsequent iteration giving more weight to a node's own current estimate and less to the influence of its neighbors.

The new updating step, replacing Equation 4 is:

$$\widehat{\mathbf{c}}_i^{(t+1)} = \beta^{(t+1)} \cdot \mathcal{M}_R(v_i^{(t)}) + (1 - \beta^{(t+1)}) \cdot \widehat{\mathbf{c}}_i^{(t)},$$

where

$$\begin{aligned} \beta^0 &= k, \\ \beta^{(t+1)} &= \beta^{(t)} \cdot \alpha, \end{aligned}$$

11. Such oscillation has been noted elsewhere for closely related methods (Murphy et al., 1999).

```
1. For $v_i \in \mathbf{V}^U$, initialize the prior: $\widehat{\mathbf{c}}_i \leftarrow \mathcal{M}_L(v_i)$, where $\widehat{\mathbf{c}}_i$ is defined as in Table 3. The
   link-based classification work of Lu and Getoor (2003) uses a local classifier to set
   initial classifications. This will clearly not work in our case (all unknowns would be
   classified as the majority class), and we therefore use a local classifier model which
   returns null (i.e., it does not return an estimation.)
2. Generate a random ordering, $O$, of elements in $\mathbf{V}^U$.
3. For elements $v_i \in O$:
   (a) Apply the relational classifier model, $\widehat{\mathbf{c}}_i \leftarrow \mathcal{M}_R$, using all non-null labels (en-
       tities which have not yet been classified will be ignored.) If all neighbor enti-
       ties are null, then return null.
   (b) Classify $v_i$:
           $x_i = c_k, k = \text{argmax}_j \widehat{\mathbf{c}}_i(j)$,
       where $\widehat{\mathbf{c}}_i(j)$ is the $j^{\text{th}}$ value in vector $\widehat{\mathbf{c}}_i$. (Remember, $\widehat{\mathbf{c}}_i$ is a vector of $|\mathcal{X}|$
       elements. We classify $v_i$ by predicting the class with the highest likelihood).
4. Repeat for $T = 1000$ iterations, or until no entities receive a new class label.[a] The
   estimates from the final iteration will be used as the final class probability estimates.
```

[a]. A post-mortem of the results showed that iterative classification often stopped within $10-20$ iterations when paired with cdRN, nBC or nLB. For wvRN, it generally ran the full 1000 iterations, although the accuracy quickly plateaued and wvRN ended up moving within a small range of similar accuracies.

Table 5: Pseudo-code for Iterative classification.

where k is a constant between 0 and 1, which for the case study we set to 1.0, and α is a decay constant, which we set to 0.99. Preliminary testing showed that final performance is very robust as long as $0.9 < \alpha < 1$. Smaller values of α can lead to neighbors losing their influence too quickly, which can hurt performance when only very few labels are known. A post-mortem of the results showed that the accuracies often converged within the first 20 iterations.

4.2.3 ITERATIVE CLASSIFICATION (IC)

The third and final collective inferencing method implemented in NetKit and used in the case study is the variant of iterative classification described in the work on link-based classification (Lu and Getoor, 2003) and shown in Table 5. As with Gibbs sampling, the relational model never sees uncertainty in the labels of (neighbor) entities. Either the label of a neighbor is `null` and ignored (which only happens in the first iteration or if all its neighbors are also `null`), or it is assigned a definite label.

5. Case Study

The study presented in this section has two goals. First, it showcases NetKit, demonstrating that the modular framework indeed facilitates the comparison of systems for learning and inference in networked data. Second, it examines the simple-but-important special case of univariate learning and inference in homogeneous networks, comparing alternative techniques that have not before been compared systematically, if at all. The setting for the case study is simple: For some entities in the network, the value of x_i is known; for others it must be estimated.

Univariate classification, albeit a simplification for many domains, is important for several reasons. First, it is a representation that is used in some applications. Above we mentioned fraud detection; as a specific example, a telephone account that calls the same numbers as a known fraudulent account (and hence the accounts are connected through these intermediary numbers) is suspicious (Fawcett and Provost, 1997; Cortes et al., 2001). For phone fraud, univariate network classification often provides alarms with reasonable coverage and remarkably low false-positive rates. Generally speaking, a homogeneous, univariate network is an inexpensive (in terms of data gathering, processing, storage) approximation of many complex networked data problems. Fraud detection applications certainly do have a variety of additional attributes of importance; nevertheless, univariate simplifications are very useful and are used in practice.

The univariate case also is important scientifically. It isolates a primary difference between networked data and non-networked data, facilitating the analysis and comparison of relevant classification and learning methods. One thesis of this study is that there is considerable information inherent in the structure of the networked data and that this information can be readily taken advantage of, using simple models, to estimate the labels of unknown entities. This thesis is tested by isolating this characteristic—namely ignoring any auxiliary attributes and only allowing the use of known class labels—and empirically evaluating how well univariate models perform in this setting on benchmark data sets.

Considering homogeneous networks plays a similar role. Although the domains we consider have obvious representations consisting of multiple entity types and edges (e.g., people and papers for node types and same-author-as and cited-by as edge types in a citation-graph domain), a homogeneous representation is much simpler. In order to assess whether a more complex representation is worthwhile, it is necessary to assess standard techniques on the simpler representation (as we do in this case study). Of course, the way a network is "homogenized" may have a considerable effect on classification performance. We will revisit this below in Section 5.3.6.

5.1 Data

The case study reported in this paper makes use of 12 benchmark data sets from four domains that have been the subject of prior study in machine learning. As this study focuses on networked data, any singleton (disconnected) entities in the data were removed. Therefore, the statistics we present may differ from those reported previously.

5.1.1 IMDB

Networked data from the Internet Movie Database (IMDb)[12] have been used to build models predicting movie success as determined by box-office receipts (Jensen and Neville, 2002a). Following the work of Neville et al. (2003), we focus on movies released in the United States between 1996 and 2001 with the goal of estimating whether the opening weekend box-office receipts "will" exceed \$2 million (Neville et al., 2003). Obtaining data from the IMDb web-site, we identified 1169 movies released between 1996 and 2001 that we were able to link up with a revenue classification

12. See `http://www.imdb.com`.

Category	**Size**
High-revenue	572
Low-revenue	597
Total	1169
Base accuracy	51.07%

Table 6: Class distribution for the IMDb data set.

Category	**Size**
Case-based	402
Genetic Algorithms	551
Neural Networks	1064
Probabilistic Methods	529
Reinforcement Learning	335
Rule Learning	230
Theory	472
Total	3583
Base accuracy	29.70%

Table 7: Class distribution for the cora data set.

in the database given to us by the authors of the original study. The class distribution of the data set is shown in Table 6.

We link movies if they share a production company, based on observations from previous work (Macskassy and Provost, 2003).[13] The weight of an edge in the resulting graph is the number of production companies two movies have in common. Notably, we ignore the temporal aspect of the movies in this study, simply labeling movies at random for the training set. This can result in a movie in the test set being released earlier than a movie in the training set.

5.1.2 CORA

The cora data set (McCallum et al., 2000) comprises computer science research papers. It includes the full citation graph as well as labels for the topic of each paper (and potentially sub- and sub-sub-topics).[14] Following a prior study (Taskar et al., 2001), we focused on papers within the machine learning topic with the classification task of predicting a paper's sub-topic (of which there are seven). The class distribution of the data set is shown in Table 7.

Papers can be linked if they share a common author, or if one cites the other. Following prior work (Lu and Getoor, 2003), we link two papers if one cites the other. The weight of an edge would normally be one unless the two papers cite each other (in which case it is two—there can be no other weight for existing edges).

13. And on a suggestion from David Jensen.
14. These labels were assigned by a naive Bayes classifier (McCallum et al., 2000).

	Number of web-pages			
Class	**Cornell**	**Texas**	**Washington**	**Wisconsin**
student	145	163	151	155
not-student	201	171	283	193
Total	346	334	434	348
Base accuracy	58.1%	51.2%	60.8%	55.5%

Table 8: Class distribution for the WebKB data set using binary class labels.

5.1.3 WEBKB

The third domain we draw from is based on the WebKB Project (Craven et al., 1998).[15] It consists of sets of web pages from four computer science departments, with each page manually labeled into 7 categories: course, department, faculty, project, staff, student, or other. As with other work (Neville et al., 2003; Lu and Getoor, 2003), we ignore pages in the "other" category except as described below.

From the WebKB data we produce eight networked data sets for within-network classification. For each of the four universities, we consider two different classification problems: the six-class problem, and following a prior study, the binary classification task of predicting whether a page belongs to a student (Neville et al., 2003).[16] The binary task results in an approximately balanced class distribution.

Following prior work on web-page classification, we link two pages by co-citations (if x links to z and y links to z, then x and y are co-citing z) (Chakrabarti et al., 1998; Lu and Getoor, 2003). To weight the link between x and y, we sum the number of hyperlinks from x to z and separately the number from y to z, and multiply these two quantities. For example, if student x has 2 edges to a group page, and a fellow student y has 3 edges to the same group page, then the weight along that path between those 2 students would be 6. This weight represents the number of possible co-citation paths between the pages. Co-citation relations are not uniquely useful to domains involving documents; for example, as mentioned above, for phone-fraud detection bandits often call the same numbers as previously identified bandits. We chose co-citations for this case study based on the prior observation that a student is more likely to have a hyperlink to her advisor or a group/project page rather than to one of her peers (Craven et al., 1998).[17]

To produce the final data sets, we extracted the pages that have at least one incoming and one outgoing link. We removed pages in the "other" category from the classification task, although they were used as "background" knowledge—allowing 2 pages to be linked by a path through an "other" page. For the binary tasks, the remaining pages were categorized into either student or not-student. The composition of the data sets is shown in Tables 8 and 9.

5.1.4 INDUSTRY CLASSIFICATION

The final domain we draw from involves classifying companies by industry sector. Companies are linked via cooccurrence in text documents. We use two different data sets, representing different

15. We use the WebKB-ILP-98 data.
16. It turns out that the relative performance of the methods is quite different on these two variants.
17. We return to these data in Section 5.3.5, where we show and discuss how using the hyperlinks directly is not sufficient for any of the univariate methods to do well.

	Number of web-pages			
Category	**cornell**	**texas**	**washington**	**wisconsin**
course	54	51	170	83
department	25	36	20	37
faculty	62	50	44	37
project	54	28	39	25
staff	6	6	10	11
student	145	163	151	155
Total	346	334	434	348
Base accuracy	41.9%	48.8%	39.2%	44.5%

Table 9: Class distribution for the WebKB data set using six-class labels.

	Number of companies	
Sector	industry-yh	industry-pr
Basic Materials	104	83
Capital Goods	83	78
Conglomerates	14	13
Consumer Cyclical	99	94
Consumer NonCyclical	60	59
Energy	71	112
Financial	170	268
Healthcare	180	279
Services	444	478
Technology	505	609
Transportation	38	47
Utilities	30	69
Total	1798	2189
Base accuracy	28.1%	27.8%

Table 10: Class distribution for the industry-yh and industry-pr data sets.

sources and distributions of documents and different time periods (which correspond to different topic distributions).

INDUSTRY CLASSIFICATION (YH)

As part of a study of activity monitoring, Fawcett and Provost (1999) collected 22,170 business news stories from the web between 4/1/1999 and 8/4/1999. Following the study by Bernstein et al. (2003), we placed an edge between two companies if they appeared together in a story. The weight of an edge is the number of such cooccurrences found in the complete corpus. The resulting network comprises 1798 companies that cooccurred with at least one other company. To classify a company, we used Yahoo!'s 12 industry sectors. Table 10 shows the details of the class memberships.

INDUSTRY CLASSIFICATION (PR)

The second industry classification data set is based on 35,318 PR Newswire press releases gathered from April 1, 2003 through September 30, 2003. As above, the companies mentioned in each press release were extracted and an edge was placed between two companies if they appeared together in a press release. The weight of an edge is the number of such cooccurrences found in the complete corpus. The resulting network comprises 2189 companies that cooccurred with at least one other company. To classify a company, we use the same classification scheme from Yahoo! as before. Table 10 shows the details of the class memberships.

5.2 Experimental Methodology

NetKit allows for any combination of a local classifier (LC), a relational classifier (RC) and a collective inferencing method (CI). If we consider an LC-RC-CI configuration to be a complete *network-classification method*, we have 12 to compare on each data set. Since, for this paper, the local classifier component is directly tied to the collective inference component (our local classifier components determine priors based on which collective inference component is being used), we henceforth consider a network-classification method to be an RC-CI configuration.

We first verify that the network structure alone (linkages plus known class labels) often contains a considerable amount of useful information for entity classification. We vary from 10% to 90% the percentage of nodes in the network for which class membership is known initially.[18] The study assesses: (1) whether the network structure enables accurate classification; (2) how much prior information is needed in order to perform well, and (3) whether there are regular patterns of improvement across methods as the percentage of initially known labels increases.

Accuracy is averaged over 10 runs. Specifically, given a data set, $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, the subset of entities with known labels $\mathbf{V}^K$ (the "training" data set[19]) is created by selecting a class-stratified random sample of $(100 \times r)\%$ of the entities in $\mathbf{V}$. The test set, $\mathbf{V}^U$, is then defined as $\mathbf{V} - \mathbf{V}^K$. We prune $\mathbf{V}^U$ by removing all nodes in *zero-knowledge* components—nodes for which there is no path to any node in $\mathbf{V}^K$. We use the same 10 training/test partitions for each network-classification method. Although it would be desirable to keep the test data disjoint (and therefore independent) as done in traditional machine learning via methods such as cross-validation, this is not applicable for within-network learning. We keep the test node sets disjoint as much as possible between the different runs. For example, at $r = 0.90$ (90% labeled data), our sets of training and testing nodes follow standard class-stratified 10-fold cross-validation.

5.3 Results

This section describes the results for our case study as well as follow-up experiments that clarify some of our findings. We start by verifying that there is information in the network structure alone (Section 5.3.1), then study the collective inference (Section 5.3.2) and relational classifier (Section 5.3.3) components separately, and then compare the network-classification methods (Section 5.3.4). We then revisit how edges were selected—we study the effects of badly chosen edges (Section 5.3.5) and then provide one method for automatic selection of edges (Section 5.3.6). We end in Section 5.3.7 with a discussion of the use of network-only methods as a baseline method that is as important as the standard non-relational-classifier baseline comparison.

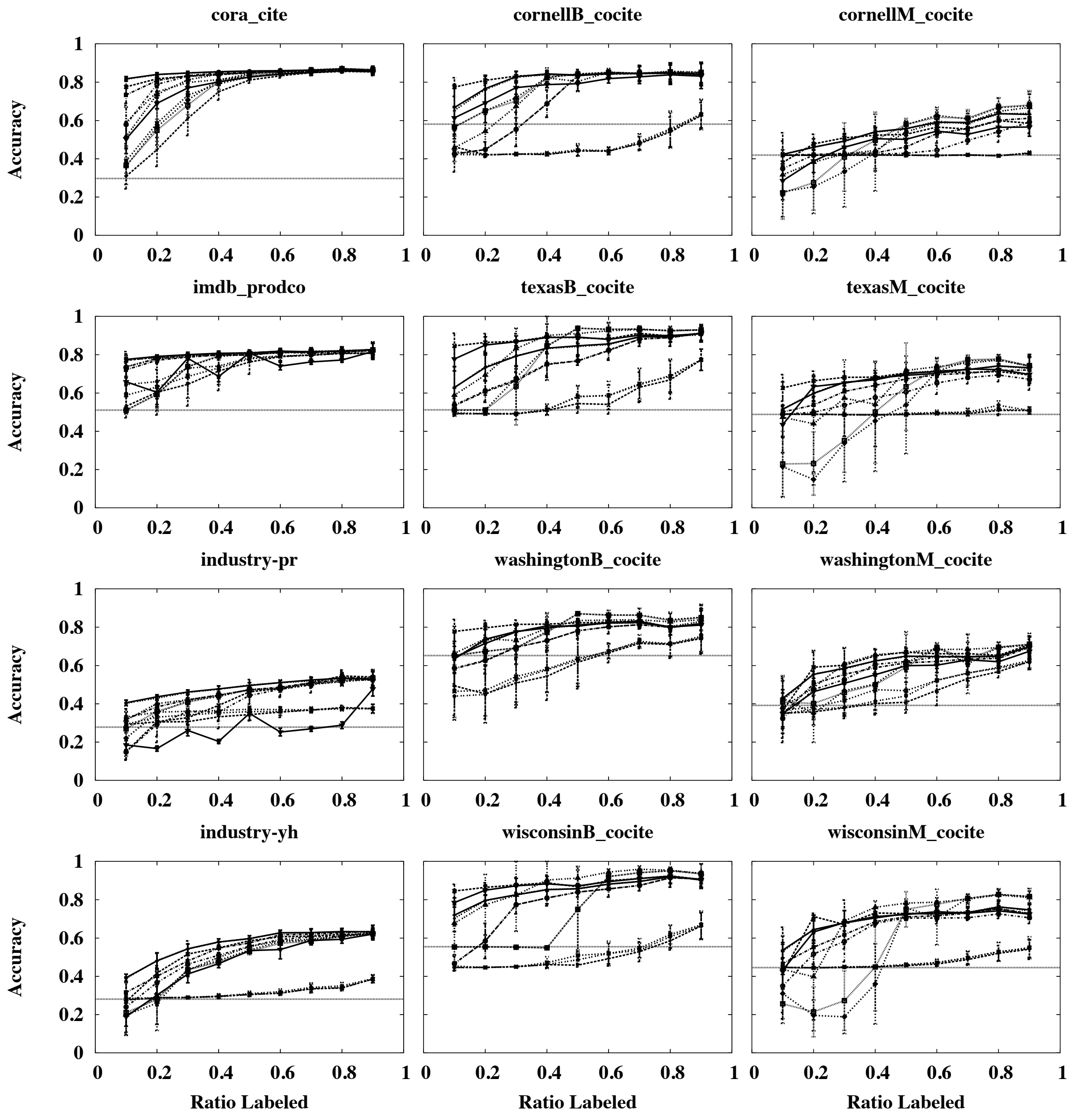


Figure 1: Overall classification accuracies on the twelve data sets. Horizontal lines represent predicting the most prevalent class. These graphs are shown to display trends and not to distinguish the twelve methods; individual methods will be clarified in subsequent discussion. The horizontal axis plots the fraction (r) of a network's nodes for which the class label is known. In each case, when many labels are known (right end) there is a set of methods that performs well. When few labels are known (left end) there is much more variation in performance. Data sets are tagged based on the edge-type used, where 'prodco' on the IMDb data is short for 'production company', and 'B' and 'M' in the WebKB data sets represent 'binary' and 'multi-class' respectively.

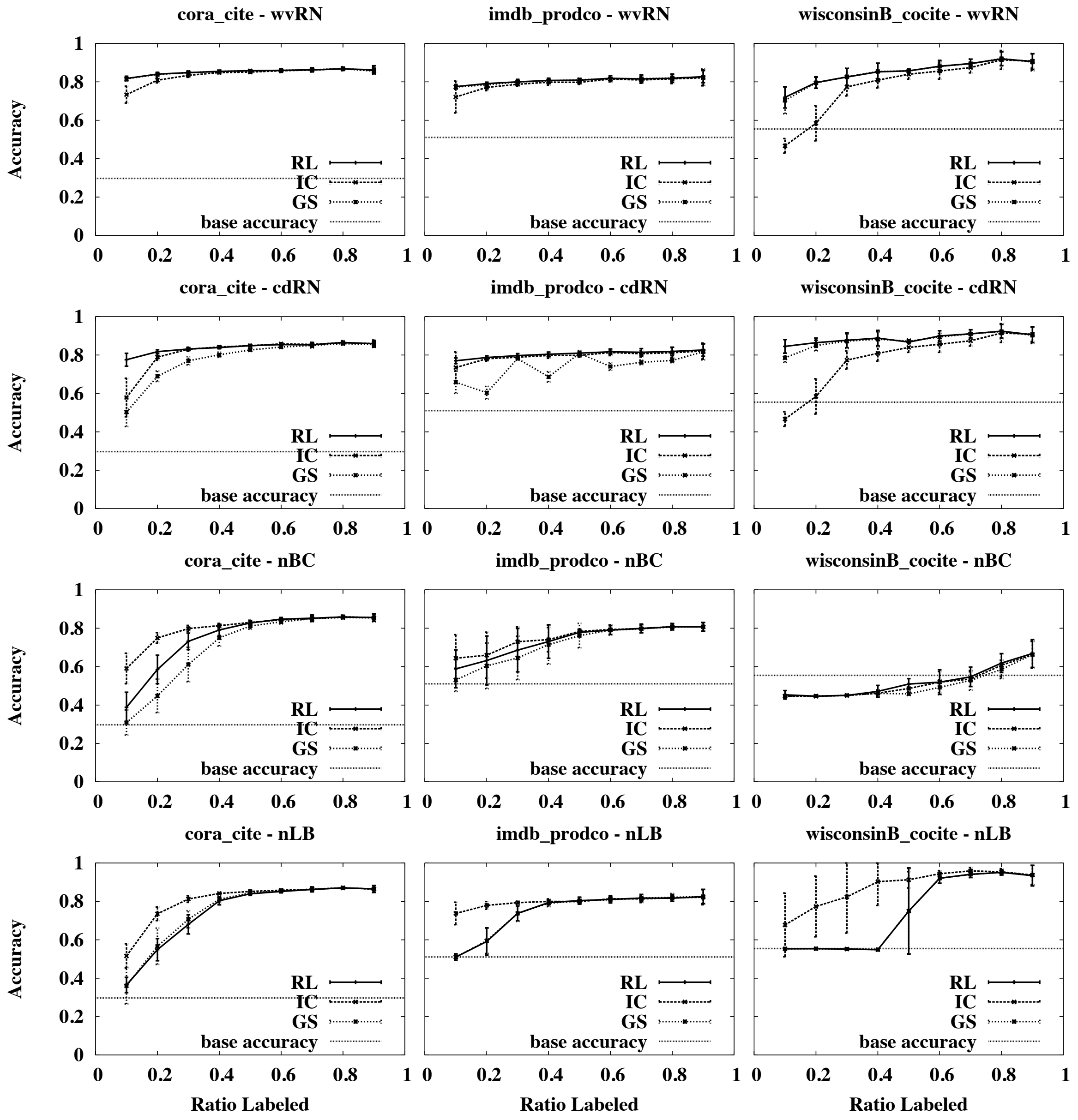


Figure 2: Comparison of collective inference methods on a few data sets. The three collective inference methods are relaxation labeling (RL), iterative classification (IC), and Gibbs sampling (GS). The data set and the relational classifier component are listed above each graph. The horizontal line represents predicting the most prevalent class.

5.3.1 INFORMATION IN THE NETWORK STRUCTURE

Figure 1 shows the accuracies of the 12 network-classification methods across the 12 data sets as the fraction (r) of entities for which class memberships are known increases from $r = 0.1$ to $r = 0.9$. The purpose of these graphs is not to differentiate the curves, but to compare the general trends and the levels of the accuracies to the baseline accuracies. In the univariate case, if the linkage structure is not known, the only non-subjective alternative is to estimate using the class base rate (prior), which is shown by the horizontal line in the graphs. As is clear from Figure 1, all of the data sets contain considerable information in the class-linkage structure. The worst relative performance is on industry-pr, where at the right end of the curves the error rate nonetheless is reduced by 30–40%. The best performance is on the binary-class WebKB-texas, where the best methods reduce the error rate by close to 90%. And in most cases, the better methods reduce the error rate by over 50% toward the right end of the curves.

Machine learning studies on networked data sets seldom have compared to simple network-classification methods like these, opting instead for comparing to non-relational classification. These results argue strongly that comparisons also should be made to univariate network classification, if the purpose is to demonstrate the power of a more sophisticated relational learning method. We return to this argument in Section 5.3.7.

5.3.2 COLLECTIVE INFERENCE COMPONENT

Figure 2 shows, for three of the data sets, the comparative performances of the three collective inference components. Each graph is for a particular relational classifier. The graphs show that while the three components often perform similarly, their performances are clearly separated for low values of r.

Table 11 shows the p-values for a Wilcoxon signed-rank test assessing whether the first method listed in column 1 is significantly better than the second. Specifically, for a given data set and label ratio (r), each network-classification experiment consists of 10 random train/test splits—the same for all configurations. For each pair of collective inference components, averaging the accuracies of the 10 splits across the 4 relational classifier components gives one average accuracy score for each of the 12 data sets yielding 12 paired data points for each collective inference method. The results show clearly that relaxation labeling, across the board, outperforms Gibbs sampling at $p \leq 0.01$ and that relaxation labeling is also better than iterative classification across the board. Further, we see that iterative classification and Gibbs sampling are roughly comparable.

The foregoing gives some evidence that relaxation labeling is consistently better than iterative classification and Gibbs sampling when the results are pooled, especially at low values of r. Table 12 quantifies the differences. In order to be comparable across data sets with different base rates, the table shows the amount of error reduction over the base rate. As a simple example, assume the base error rate is 0.4, method A yields an error rate of 0.1, and method B yields an error rate of 0.2. Method A reduces the error by 75%. Method B reduces the error by 50%.

18. We also performed boundary experiments using the weighted-vote relational neighbor classifier with relaxation labeling, where we decreased the number of initially known labels down to 0.1% of the graph. The classification accuracy generally degrades gracefully to doing no better than random guessing at this extreme.
19. These data will be used not only for training models, but also as existing background knowledge during classification.

	sample ratio (r)								
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
RL v *GS*	**0.002**	**0.001**	**0.001**	**0.001**	**0.002**	**0.001**	**0.001**	**0.010**	**0.010**
RL v *IC*	**0.025**	**0.100**	**0.100**	**0.025**	**0.002**	**0.002**	**0.001**	**0.005**	**0.002**
IC v *GS*	*0.400*	**0.450**	—	**0.450**	*0.400*	—	*0.250*	—	**0.400**

Table 11: p-values for the Wilcoxon signed-rank test, comparing the accuracies of pairs of three collective inference methods across all data sets and relational classifier methods. (The three collective inference methods are relaxation labeling (RL), iterative classification (IC), and Gibbs sampling (GS)). For each cell, bold text means that the first method was better than the second method and italics means it was worse. The cells with '—' means there was no significant difference.

	sample ratio (r)									
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**overall**
RL	**0.081**	**0.184**	**0.262**	**0.339**	**0.409**	**0.446**	**0.468**	**0.491**	**0.509**	**0.354**
IC	0.000	0.116	0.235	0.314	0.382	0.423	0.452	0.482	0.503	0.323
GS	0.029	0.126	0.231	0.301	0.385	0.421	0.450	0.474	0.502	0.324

Table 12: Relative error reductions (ER_{REL}) for each collective inference method across all data sets and relational classifier methods. (The three collective inference methods are relaxation labeling (RL), iterative classification (IC), and Gibbs sampling (GS)). The last column, **overall**, is the average error reduction for each method across all sample ratios. Bold entries indicate the largest relative error reduction for each sample ratio.

	sample ratio (r)									
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**total**
RL	11	11	10	7	4	5	6	4	6	64
GS	1	1	0	1	5	3	4	4	4	23
IC	0	0	2	4	3	4	2	4	2	21

Table 13: Number of times each collective inference method was the best across the 12 data sets. The three methods are relaxation labeling (RL), iterative classification (IC), and Gibbs sampling (GS).

The relative error reduction of the collective inference (CI) components is defined as follows:

$$\begin{aligned}
\mathrm{ER}_{\mathrm{ABS}}(\mathrm{RC},\mathrm{CI},D,r) &= (\mathrm{base\_err}(D) - \mathrm{err}(\mathrm{RC}\text{-}\mathrm{CI},D,r)), \\
\mathrm{ER}_{\mathrm{REL}}(\mathrm{RC},\mathrm{CI},D,r) &= \frac{\mathrm{ER}_{\mathrm{ABS}}(\mathrm{RC},\mathrm{CI},D,r)}{\mathrm{base\_err}(D)}, \\
\mathrm{ER}_{\mathrm{REL}}(\mathrm{RC},\mathrm{CI},r) &= \frac{1}{|\mathbf{D}|}\sum_{D\in\mathbf{D}} \mathrm{ER}_{\mathrm{REL}}(\mathrm{RC},\mathrm{CI},D,r), \\
\mathrm{ER}_{\mathrm{REL}}(\mathrm{CI},r) &= \frac{1}{|\mathbf{RC}|}\sum_{\mathrm{RC}\in\mathbf{RC}} \mathrm{ER}_{\mathrm{REL}}(\mathrm{RC},\mathrm{CI},r),
\end{aligned}$$

where err(RC-CI,D,r) is the error for the RC-CI configuration (a relational classifier paired with a collective inference method) on data set D with r% of the graph being labeled.

Table 12 shows the relative error reductions for each collective inference component. Relaxation labeling outperformed iterative classification across the board, from as low as a 1.3% improvement ($r = 0.90$) to as high as 59% or better improvement ($r \leq 0.2$) when averaged over all the data sets and relational classifier methods. Overall relaxation labeling improved performance over iterative classification by about 10% as seen in the last column. Notably, relaxation labeling's advantage over iterative classification improves monotonically as less is known in the network. Similar numbers and a similar pattern are seen for relaxation labeling versus Gibbs sampling. Iterative classification and Gibbs sampling are comparable.

For another perspective, Table 13 shows, for each ratio as well as a total across all ratios, the number of times each collective inference implementation took part in the best-performing network-classification combination for each of the 12 data sets. Specifically, for each sampling ratio, each win for an RC-CI configuration (relational classifier paired with a collective inference method) counted as a win for the collective inference module of the pair (as well as a win for the relational classifier module in the next section). For example, in Figure 2, the first column of four graphs shows the performances of the 12 network-classification combinations on the cora data; at the left end of the curves, wvRN-RL is the best performing combination. Table 13 adds further support to the conclusion that relaxation labeling was the overall best component, primarily due to its advantage at low values of r. We also see again that Gibbs sampling and iterative classification were comparable.

5.3.3 RELATIONAL CLASSIFIER COMPONENT

Comparing relational models, we would expect to see a certain pattern: if even moderate homophily is present in the data, we would expect wvRN to perform well. Its nonexistent training variance[20] should allow it to perform relatively well, even with small numbers of known labels in the network. The higher-variance nLB may perform relatively poorly with small numbers of known labels (primarily because of the lack of training data). On the other hand, wvRN is potentially a very-high-bias classifier—it does not learn a relational model at all. The learning-based classifiers may well perform better with large numbers of known labels if there are patterns beyond homophily to be learned. As a worst case for wvRN, consider a bipartite graph between two classes. In a leave-one-out cross-validation, wvRN would be wrong on every prediction. The relational learners should notice the true pattern immediately. More generally, one should expect a pattern of curves crossing with increasing numbers of known labels, similar to learning curves crossing (Perlich et al., 2003): lower variance methods (such as wvRN) should be preferable with fewer known labels; more flexible learning methods should be preferable with more known labels (thus, more training data).

Figure 3 shows for four of the data sets the performances of the four relational classifier implementations. The rows of graphs correspond to data sets and the columns to the three different collective inference methods. The graphs show several things. As would be expected, accuracy improves as more of the network is labeled, although in certain cases classification is remarkably accurate with very few known labels (e.g., see cora). One method is substantially worse than the others. Among the remaining methods, performance often differs greatly with few known labels, and tends to converge with many known labels. More subtly, as expected there often is a crossing of curves when about half the nodes are labeled (e.g., see Washington).

20. There still will be variance due to the set of known labels.

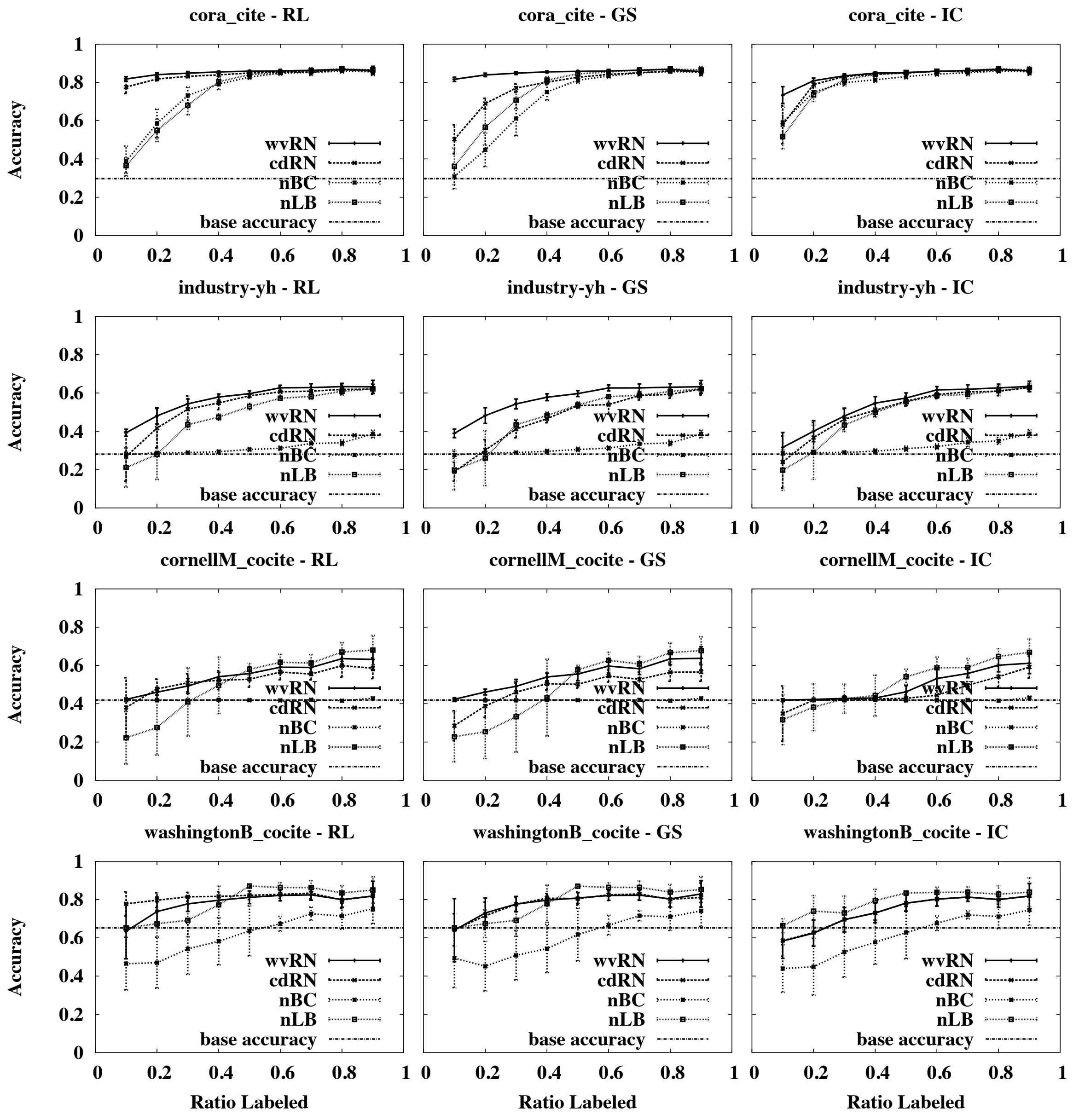


Figure 3: Comparison of the relational classifiers on a few data sets. The data set (and link-type) and the paired collective inference component are listed above each graph. (The three collective inference methods are relaxation labeling (RL), iterative classification (IC), and Gibbs sampling (GS)). The horizontal line represents predicting the most prevalent class.

	sample ratio (r)								
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
wvRN v *cdRN*	**0.025**	**0.200**	**0.050**	**0.100**	**0.025**	**0.025**	**0.010**	**0.002**	**0.001**
wvRN v *nBC*	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**
wvRN v *nLB*	**0.001**	**0.001**	**0.001**	**0.005**	*0.400*	*0.005*	*0.002*	*0.001*	*0.001*
cdRN v *nBC*	**0.050**	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**
cdRN v *nLB*	**0.010**	**0.001**	**0.010**	**0.200**	*0.050*	*0.001*	*0.001*	*0.001*	*0.001*
nLB v *nBC*	—	**0.200**	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**

Table 14: p-values for the Wilcoxon signed-rank test, comparing the accuracies between pairs of relational classifiers across all data sets and collective inference methods. For each cell, bold text means that the first method was better than the second method and italic text means it was worse. The cells with '—' showed no significant difference.

	sample ratio (r)									
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**overall**
cdRN	0.106	0.273	0.370	0.423	0.474	0.496	0.519	0.533	0.549	0.416
nLB	−0.049	0.039	0.206	0.354	**0.508**	**0.573**	**0.587**	**0.609**	**0.609**	0.382
wvRN	**0.157**	**0.299**	**0.392**	**0.453**	0.494	0.525	0.543	0.563	0.571	**0.444**

Table 15: Relative error reductions (ER_{REL}) for each relational classifier component across all data sets. Each cell shows the error reduction of the given method. The last column, **overall**, is the average error reduction for the methods across all sample ratios. Bold entries indicate the largest relative error reduction for each sample ratio.

	sample ratio (r)									
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**total**
wvRN	7	4	4	6	4	4	2	1	2	34
cdRN	5	8	6	2	1	0	0	1	1	24
nLB	0	0	2	4	7	8	10	10	9	50

Table 16: Number of times each relational classifier implementation was the best across the 12 data sets.

Table 14 shows statistical significance results, computed as described in the previous section (except here varying the relational classifier component). nBC was almost always significantly and often substantially worse than the other three relational classifiers and therefore for clarity is eliminated from the remainder of this analysis. Although cdRN and wvRN were close, wvRN was generally better than cdRN. Examining the wvRN and cdRN methods versus nLB we see the same pattern: at $r = 0.5$, the advantage shifts from the wvRN/cdRN methods to nLB, consistent with our expectations discussed above.

Table 15 shows the error reductions for each relational classifier comparison, computed as in the previous section with the obvious changes between relational classifier and collective inference.

The same patterns are evident as observed from Table 14. Further, we see that the differences can be large: when the wvRN/cdRN methods are better, they often are much better. The link-based (nLB) classifier also can be considerably better than wvRN/cdRN.

Table 16 shows how often each relational classifier method participated in the best combination, as described in the previous section. nLB is the overall winner, but we see the same clear pattern that the wvRN/cdRN methods dominate for fewer labels, and nLB dominates for more labels, with the advantage changing hands at $r = 0.5$.

5.3.4 INTERACTION BETWEEN COMPONENTS

Table 17 shows how many times each of the 9 individual relational classifier/collective inference (RC-CI) configurations was the best, across the 12 data sets and 9 labeling ratios.[21] Strikingly, two sets of closely related configurations stand out: wvRN-RL and cdRN-RL; and nLB with any of the collective inference methods. The wvRN-RL/cdRN-RL methods dominate for $r < 0.5$; the nLB methods (nLB paired with any collective inference method) dominate for $r \geq 0.5$. Table 18 and Table 19 compare these five methods analogously to the previous sections. (As before, each cell comprises 12 data points, each being an average accuracy of one data set over 10 repetitions.) The clear pattern is in line with that shown in the prior sections, showing that of this set of best methods, the wvRN/cdRN methods excel for fewer labeled data, and the nLB-based method excels for more labeled data.

In addition, these results show that the wvRN-/cdRN-methods clearly should be paired with relaxation labeling, possibly because RL allows them to average estimates between zero and one, resulting in lower variance, especially with few known labels. nLB, on the other hand, does not favor one collective inference method over the others.

Following up on these results, a 2-way ANOVA shows a strong interaction between the relational classifier and collective inference components for most data sets for small numbers of labeled nodes, as would be expected given the strong performance of the specific pairings wvRN-RL and cdRN-RL. As more nodes are labeled, the interaction becomes insignificant for almost all data sets, as might be expected given that nLB dominates but no collective inference method does. The ANOVA adds weight to the conclusion that for very many known labels, it matters little which collective inference method is used for accuracy maximization, so perhaps the most efficient method can be chosen.

5.3.5 THE IMPORTANCE OF EDGE SELECTION

To create homogeneous graphs, we had to select the edges to use. As mentioned briefly above, the type of edge selected can have a substantial impact on classification accuracy. For these data sets, the worst case (we have found) occurs for WebKB. As described in Section 5.1.3, for the results presented so far we have used co-citation links, based on observations in prior published work. An obvious alternative is to use the hyperlinks themselves.

Figures 4 and 5 compare the results using hyperlinks with those using co-citation links. Using hyperlinks, the network-classification methods perform much worse than with co-citations. Although there is some lift at large values of r, especially for the Washington data, the performance is not comparable to that with the co-citation formulation. The transformation from the hyperlink-

21. As mentioned before, we are no longer including nBC in the comparisons. As it turns out, nBC did not have any wins, regardless of the collective inference method it was paired with.

	sample ratio (r)									
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**total**
wvRN-IC	0	0	0	0	0	0	0	0	1	1
wvRN-GS	1	1	0	1	3	0	0	0	0	6
wvRN-RL	6	3	4	5	1	4	2	1	1	27
cdRN-IC	0	0	0	0	0	0	0	0	0	0
cdRN-GS	0	0	0	0	0	0	0	0	0	0
cdRN-RL	5	8	6	2	1	0	0	1	1	24
nLB-IC	0	0	2	4	3	4	2	4	1	20
nLB-GS	0	0	0	0	2	3	4	4	4	17
nLB-RL	0	0	0	0	2	1	4	2	4	13

Table 17: Number of times each relational classifier and collective inference configuration was the best across the 12 data sets. The three collective inference methods are relaxation labeling (RL), iterative classification (IC), and Gibbs sampling (GS).

	sample ratio (r)								
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
wvRN-RL v *cdRN-RL*	*0.450*	*0.200*	*0.250*	*0.400*	—	**0.450**	**0.400**	**0.200**	**0.100**
wvRN-RL v *nLB-GS*	**0.010**	**0.010**	**0.010**	**0.010**	**0.300**	*0.250*	*0.100*	*0.025*	*0.025*
wvRN-RL v *nLB-IC*	**0.010**	**0.010**	**0.050**	**0.300**	*0.300*	*0.200*	*0.200*	*0.050*	*0.025*
wvRN-RL v *nLB-RL*	**0.010**	**0.010**	**0.010**	**0.010**	**0.250**	*0.200*	*0.100*	*0.025*	*0.020*
cdRN-RL v *nLB-GS*	**0.010**	**0.010**	**0.010**	**0.020**	**0.400**	*0.200*	*0.100*	*0.050*	*0.025*
cdRN-RL v *nLB-IC*	**0.050**	**0.020**	**0.050**	**0.300**	*0.200*	*0.025*	*0.050*	*0.025*	*0.025*
cdRN-RL v *nLB-RL*	**0.010**	**0.010**	**0.010**	**0.025**	**0.400**	*0.050*	*0.100*	*0.025*	*0.020*
nLB-IC v *nLB-GS*	**0.050**	**0.010**	**0.020**	**0.010**	**0.400**	**0.400**	*0.300*	*0.300*	*0.400*
nLB-IC v *nLB-RL*	**0.050**	**0.025**	**0.020**	**0.025**	**0.400**	**0.400**	*0.250*	*0.200*	*0.100*
nLB-RL v *nLB-GS*	**0.450**	**0.100**	—	**0.300**	*0.200*	**0.400**	**0.200**	**0.300**	**0.100**

Table 18: *p*-values for the Wilcoxon signed-rank test, comparing the accuracies among the five best relational classifier/collective inference configurations across all data sets. (The three collective inference methods are relaxation labeling (RL), iterative classification (IC), and Gibbs sampling (GS)). For each cell, bold text means that the first method was better than the second method and italic text means it was worse. The cells with '—' showed no significant difference.

	sample ratio (r)									
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**overall**
wvRN-RL	0.222	0.369	0.444	0.489	0.514	0.537	0.551	0.568	0.573	0.474
cdRN-RL	**0.274**	**0.430**	**0.462**	**0.501**	0.519	0.538	0.549	0.556	0.558	**0.488**
nLB-RL	−0.101	−0.024	0.131	0.311	0.499	**0.577**	**0.590**	**0.611**	**0.612**	0.356
nLB-IC	0.054	0.191	0.367	0.468	**0.535**	0.575	0.587	0.606	0.606	0.443
nLB-GS	−0.100	−0.051	0.121	0.284	0.489	0.567	0.585	**0.611**	0.610	0.346

Table 19: Relative error reduction (ER_{REL}) improvements for the 5 best relational classifier/collective inference configurations across all data sets. (The three collective inference methods are relaxation labeling (RL), iterative classification (IC), and Gibbs sampling (GS)). Each cell shows the error reduction of the given method. The last column, **overall**, is the average error reduction for the methods across all sample ratios. Bold entries indicate the largest relative error reduction for each sample ratio.

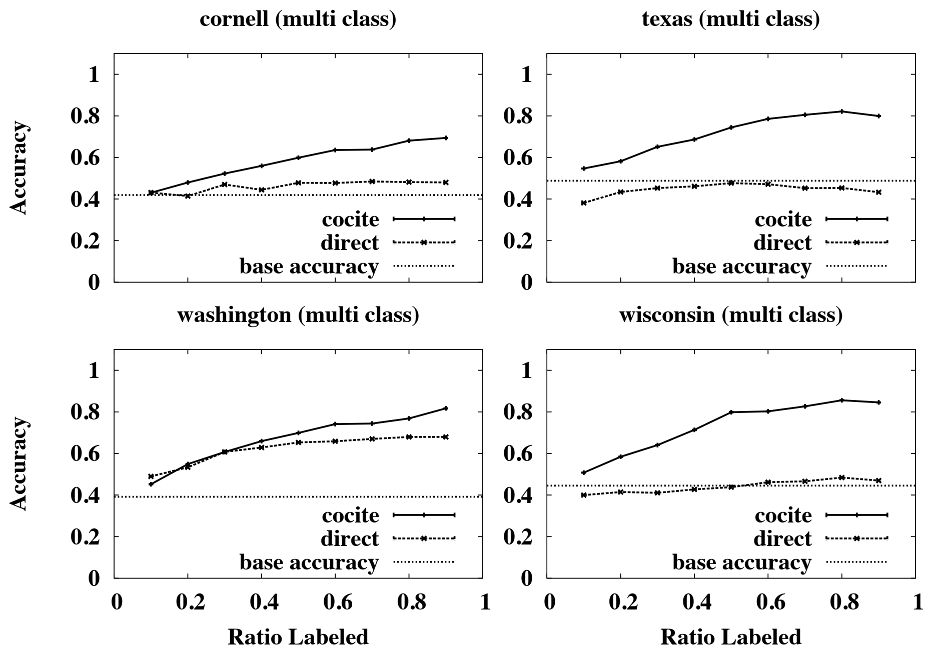


Figure 4: Comparative performances on WebKB multi-class problems using co-citation links versus using direct hyperlinks as edges. We show here the average performance of the 12 network learners.

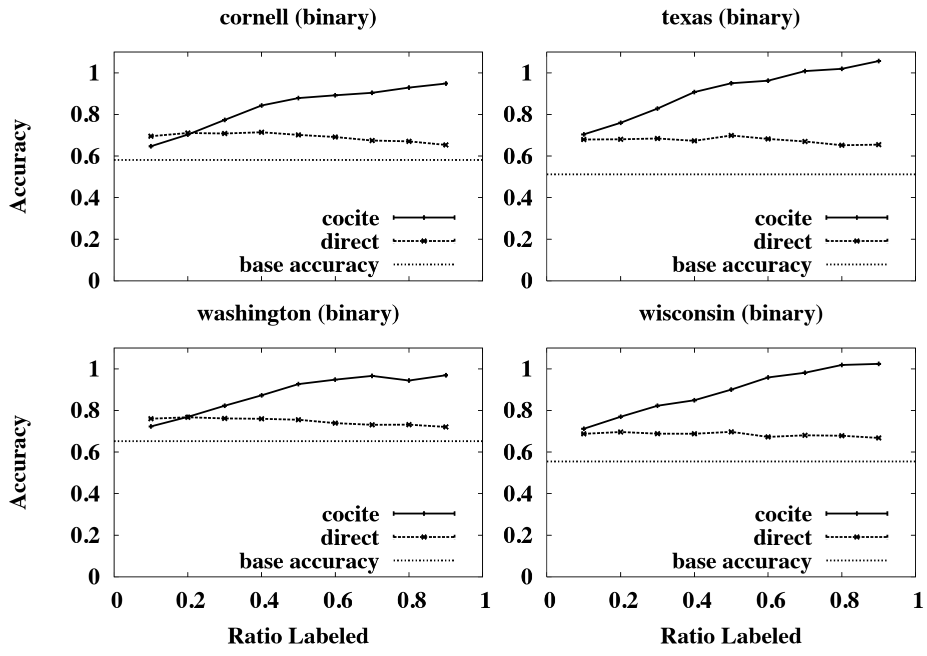


Figure 5: Comparative performances on WebKB binary-class problems using co-citation links versus using direct hyperlinks as edges. We show here the average performance of the 12 network learners.

based network to the co-citation-based network adds no new information to the graph. However, in the hyperlink formulation the network classification methods cannot take full advantage of the information present—mainly because of the stark violation of the first-order Markov assumption made by the relational classifiers. These results demonstrate that the choice of edges can be crucial for good performance.

5.3.6 AUTOMATIC EDGE SELECTION

Creating a graph with a single type of edge from a problem where various possible links exist is a representation engineering problem reminiscent of the selection of a small set of useful features for traditional classification.[22] For feature selection, practitioners use a combination of domain knowledge, analysis, and trial and error to select a good representation. To create the networked data for our study, we chose edges based on suggestions from prior work—which indirectly combines domain knowledge, prior analysis and prior trial and error, although we explicitly avoided choosing the representations based on their performance using NetKit.

Pursuing the analogy with choosing features, it may be possible to select edges automatically. It is beyond the scope of this paper to address the general (open and important) problem of edge selection; however, the excellence (on these data sets) and simplicity of wvRN suggests straightforward techniques.

If we consider the data sets used in the study, all but the industry classification data sets have more than one reasonable choice for defining the edges:

1. **cora**: We linked entities through citations (*cite*). Alternatively, we could have linked by the sharing of an author (*author*), or by either relationship (combined as a single generic link).

2. **imdb**: There are many types of ways to connect two movies, but we focus here on four that were suggested to us by David Jensen: *actor*, *director*, *producer* and production company (*prodco*). Again, we could use any or all of them (we do not consider all possible combinations here).

3. **WebKB**: Based on prior work, we chose co-citations (*cocite*) for the case study and later showed that the original hyperlinks (*hyper*) were a poor choice.

Kohavi and John (1997) differentiate between wrapper approaches and filter approaches to feature selection, and this notion extends directly to edge selection. For any network classification method we can take a wrapper approach, computing the error reduction over $\mathbf{G}^K$ using cross-validation. wvRN is an attractive candidate for such an approach, because it is very fast and requires no training; we can use a simple leave-one-out (loo) estimation.

The homophily-based wvRN also lends itself to a filter approach, selecting the edge type simply by measuring the homophily in $\mathbf{G}^K$. Heckathorn and Jeffri (2003) define a homophily index, but it computes homophily for a specific group, or class, rather than a general value across all classes. The *assortativity coefficient* (Newman, 2003) is based on the correlation between the classes linked by edges in a graph. Specifically, it is based on the graph's assortativity matrix—a CxC matrix, where

22. We required a single edge type for our homogeneous case study; it is reasonable to conjecture that even if heterogeneous links are allowed, a small set of good links would be preferable. For example, a link-based classifier produces a feature-vector representation with multiple positions per link type.

cell e_{ij} represents the fraction of (all) edges that link nodes of class c_i to nodes of class c_j, such that $\sum_{ij} e_{ij} = 1$. The assortativity coefficient, A_E, is calculated as follows:

$$\begin{aligned} a_i &= \sum_j e_{ij}, \\ b_j &= \sum_i e_{ij}, \\ A_E &= \frac{\sum_i e_{ii} - \sum_i a_i \cdot b_i}{1 - \sum_i a_i \cdot b_i}. \end{aligned}$$

However, A_E measures homophily across edges, while wvRN is based on homophily across nodes. It is possible to create (sometimes weird) graphs with high A_E but for which wvRN performs poorly, and vice versa. However, we can modify A_E to be a *node-based assortativity* coefficient, A_N, by defining e^*_{ij}, a node-based cell-value in the assortativity matrix as follows:

$$e^*_{ij} = \frac{1}{Z}\text{RV}(X_i)_j,$$

where $\text{RV}(X_i)_j$ is the j^{th} element in $\text{RV}(X_i)$ as defined in Equation 2, and Z is a normalizing constant such that all e_{ij} sum to 1.

To assess their value for edge selection for wvRN, we compute the actual error reduction for each different edge type (and all edges) for the benchmark data sets, and compare with the edge selected by each of these three methods (loo, A_E, A_N). In Table 20 the first six columns show the data set, the number of nodes, the base accuracy, the number of edges, the average edge weight, and the average node degree. The next columns show A_E and A_N. The following column shows the estimated $\widehat{\text{ER}}_{\text{REL}}$ based on the leave-one-out estimation, and the last column shows the ER_{REL} values on the test set. Each data set group is sorted by the ER_{REL} performance on its various edge types, so the top row of each group is the "best" edge selection. Note that as the edge types differ, we get different connectivities and different coverages, and hence different the values are not completely comparable.

The results show that the links used in our study generally resulted in the highest node-based assortativity.[23] In 8 out of 10 cases, A_N chose the best edge. Neither the leave-one-out (loo) method nor A_E performed as well, but they nevertheless yield networks on which wvRN performs relatively well. Notice that for IMDb, although director has the highest A_E, it also has very low coverage (only 554 nodes were connected), and with such a slight difference in assortativity between that and prodco there should be no question which should be used for classification. A_N and the leave-one-out estimates are much more volatile than A_E as the amount of labeled data decreases, because typically there are many more edges than nodes. If we believe that assortativity is relatively stable across the network, it may be beneficial to use A_E when little is known. However, for our data sets, A_N performs just as well as A_E even when $r = 0.1$.

In summary, these results show that the differences in the results by domain, and even within domains, depend on the level of homophily. This is almost tautological for wvRN. We also computed many other graph statistics besides the homophily measures (e.g., average degree, number of connected clusters, graph diameter, shortest paths, etc.) none of which were as strongly correlated with performance.

23. We had picked the edge types for the study before performing this analysis. However, common sense and domain knowledge would lead one to conclude that the edge types we used in the case study would have high assortativity.

Data set	size	base acc.	num edges	mean edge weight	mean node degree	Assortativity (edge) A_E	(node) A_N	$\widehat{\text{ER}}_{\text{REL}}$ loo (wvRN) $r=0.90$	ER_{REL} wvRN at $r=0.90$
$\text{cora}_{\text{cite}}$	3583	0.297	22516	2.061	6.284	**0.737**	0.642	0.5373	0.805
cora_{all}	4025	0.315	71824	2.418	17.844	0.656	**0.656**	**0.6122**	0.767
$\text{cora}_{\text{author}}$	3604	0.317	56268	2.262	15.613	0.623	0.558	0.4662	0.711
$\text{imdb}_{\text{prodco}}$	1169	0.511	40634	1.077	34.760	0.501	**0.392**	**0.3711**	0.647
$\text{imdb}_{\text{producers}}$	1195	0.520	13148	1.598	11.003	0.283	0.389	0.3618	0.547
imdb_{all}	1377	0.564	92248	1.307	66.992	0.279	0.308	0.3415	0.531
$\text{imdb}_{\text{directors}}$	554	0.549	826	1.031	1.491	**0.503**	0.210	0.0369	0.498
$\text{imdb}_{\text{actors}}$	1285	0.541	48354	1.135	37.630	0.131	0.174	0.1372	0.246
$\text{cornellB}_{\text{all}}$	349	0.585	27539	3.000	78.908	0.325	**0.399**	**0.5655**	0.629
$\text{cornellB}_{\text{cocite}}$	346	0.581	26832	2.974	77.549	**0.360**	0.394	0.5345	0.618
$\text{cornellB}_{\text{hyper}}$	349	0.585	1393	2.349	3.991	−0.169	−0.068	−0.1621	−0.114
$\text{cornellM}_{\text{all}}$	349	0.415	27539	3.000	78.908	0.219	**0.286**	**0.3209**	0.382
$\text{cornellM}_{\text{cocite}}$	346	0.419	26832	2.974	77.549	**0.227**	0.273	0.2481	0.366
$\text{cornellM}_{\text{hyper}}$	349	0.415	1393	2.349	3.991	0.054	0.102	−0.2883	−0.212
$\text{texasB}_{\text{cocite}}$	334	0.512	32988	2.961	98.766	**0.577**	**0.617**	**0.7166**	0.819
$\text{texasB}_{\text{all}}$	338	0.518	33364	2.995	98.710	0.523	0.585	0.6939	0.768
$\text{texasB}_{\text{hyper}}$	285	0.547	1001	2.605	3.512	−0.179	−0.114	−0.1368	−0.232
$\text{texasM}_{\text{cocite}}$	334	0.488	32988	2.961	98.766	**0.461**	**0.477**	0.3737	0.475
$\text{texasM}_{\text{all}}$	338	0.482	33364	2.995	98.710	0.420	0.458	**0.3874**	0.466
$\text{texasM}_{\text{hyper}}$	285	0.453	1001	2.605	3.512	−0.033	−0.044	−0.6583	−0.490
$\text{washingtonB}_{\text{all}}$	434	0.652	31253	3.800	72.012	**0.388**	**0.455**	**0.4225**	0.530
$\text{washingtonB}_{\text{cocite}}$	434	0.652	30462	3.773	70.189	0.375	0.446	0.3940	0.477
$\text{washingtonB}_{\text{hyper}}$	433	0.651	1941	2.374	4.483	−0.095	0.076	−0.1126	−0.069
$\text{washingtonM}_{\text{cocite}}$	434	0.392	30462	3.773	70.189	0.301	0.359	0.3481	0.503
$\text{washingtonM}_{\text{all}}$	434	0.392	31253	3.800	72.012	**0.331**	**0.377**	**0.4023**	0.453
$\text{washingtonM}_{\text{hyper}}$	433	0.390	1941	2.374	4.483	0.084	0.233	−0.0167	0.004
$\text{wisconsinB}_{\text{all}}$	352	0.560	33587	3.543	95.418	0.524	**0.587**	**0.7219**	0.855
$\text{wisconsinB}_{\text{cocite}}$	348	0.555	33250	3.499	95.546	**0.673**	0.585	0.7168	0.788
$\text{wisconsinB}_{\text{hyper}}$	297	0.616	1152	2.500	3.879	−0.147	−0.103	−0.2123	−0.331
$\text{wisconsinM}_{\text{cocite}}$	348	0.445	33250	3.499	95.546	**0.577**	**0.489**	0.4286	0.544
$\text{wisconsinM}_{\text{all}}$	352	0.440	33587	3.543	95.418	0.416	0.474	**0.4518**	0.503
$\text{wisconsinM}_{\text{hyper}}$	297	0.384	1152	2.500	3.879	0.160	0.021	−0.4729	−0.275
# mistakes						5	2	4	

Table 20: Graph metrics for networks formed by various edge types. Each data set grouping is sorted on ER_{REL}. A_E, A_N, $\widehat{\text{ER}}_{\text{REL}}$ and ER_{REL} values were all averaged over the 10 data splits used throughout the case study. The leave-one-out (loo) measure used only $\mathbf{G}^K$ to calculate the $\widehat{\text{ER}}_{\text{REL}}$ value.

5.3.7 NETWORK-ONLY CLASSIFICATION AS A BASELINE FOR RELATIONAL LEARNING STUDIES

The foregoing results support the claim that network-only classification should be used as a baseline against which more sophisticated network-classification methods should be compared. Therefore, for example, when evaluating a system for the classification of interconnected web pages, their connectivity alone should be considered in addition to their text alone. This of course is sound scientific methodology, requiring one to look for simpler explanations for a phenomenon once a complex explanation has been found. When considering a sophisticated relational learning method on networked data, using the network alone is one sort of lesion or ablation study (Langley, 2000). Our point here is that the absolute classification performance based only on the network of class labels is remarkable in several of the cases examined above. For example, who would have thought that on the University of Texas student/not-student classification task, accuracy could be increased from 62.9% to 91.2% using just the network structure—with no text and no learning?

Nevertheless, it is enlightening to compare these simple methods to alternatives. In this section we provide several such comparisons. We compare to text-only methods and to a relational learning technique that uses both local and relational information, and we see that one might be led to be overly optimistic by comparing solely to local-only baselines. We also compare to a graph-based method that does not fall into the "node-centric" framework. Finally, we replicate a prior, published study, showing that conclusions would have changed if a network-only baseline had been used.

COMPARISON TO A MULTIVARIATE TECHNIQUE

In real networks, the amount of local information and its relevance to the classification task vary widely. In certain social network classification applications, for example in support of counterterrorism, no information at all may be available about certain individuals other than their connections to others (Tumulty, 2006). In applications such as fraud detection in telecommunications little reliable information may be available about potential fraudsters, beyond the calls they make (or as discussed above, local information may be ignored for many individuals because network-only classification works so well). On the other hand, the text of certain web pages contains a wealth of information relevant for classifying the topic of the web page.

For determining how much better can be done by using both the text and the interconnectivity information, even for existing techniques and benchmark data sets, a definitive answer would involve a comparison of a large number of alternative relational learning systems on equal experimental footing. Such a comparison is beyond the scope of this paper; to our knowledge, no such comparison has yet been attempted, but one would be facilitated by a platform such as NetKit. To demonstrate, we compare to a reimplementation in NetKit of the highly cited approach by Chakrabarti et al. (1998), which we will call NetKit-CDI.

Thus, for those of our data sets that contained local information, we compared the accuracies using only the local information with the accuracies using only the network as well as both relational and local information.[24] It must be noted that even within these data sets, there are nodes that have no local information, just as there are nodes with no connectivity information.[25] We compare classification accuracies only for those nodes that have both local information and connectivity information. Specifically, the local information is text, and for local classification we use naive

24. We have no local information on the IMDb data or the industry-yh data.
25. The industry-pr and WebKB data sets only have local information for roughly half the instances.

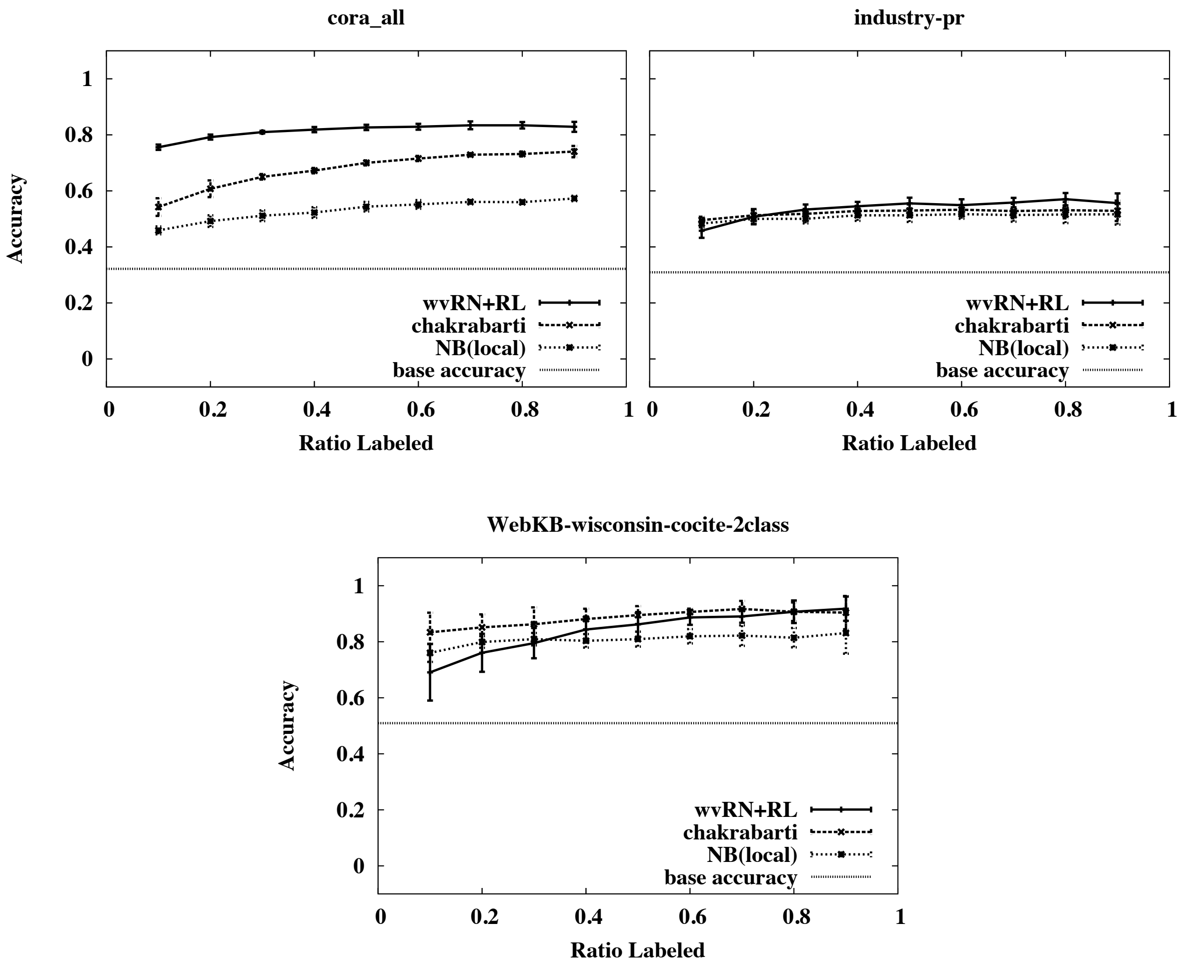


Figure 6: Comparisons of wvRN-RL (relaxation labeling), a reimplementation of the method of Chakrabarti et al. (NetKit-CDI), and a local-only naive Bayes classifier (NB(local)).

Bayes classification (which in preliminary studies, when averaged over the data sets, performed better than Rocchio's algorithm and multinomial Naive Bayes).

Figure 6 shows the accuracies on three of the ten data sets. We show these three to highlight the variance in relative performances, making it clear that neither network-only nor local-only should be the sole baseline of choice. We performed the same statistical analyses of these results as we did in the main study. Table 21 shows the p-values of the Wilcoxon signed-rank test across the ten data sets. The table shows that no method is significantly better than any other at $r = 0.1$, but the network-only is significantly better than both the full relational classifier and the local-only for $r \geq 0.3$ ($p \leq 0.100$) and $r \geq 0.4$ ($p \leq 0.025$ except against NetKit-CDI at $r = 0.9$). We also found that the relational classifier generally performed better than the local-only method at $r \geq 0.4$,

	sample ratio (r)								
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
wvRN-RL v *NB-local*	*0.200*	—	**0.100**	**0.025**	**0.025**	**0.025**	**0.025**	**0.025**	**0.025**
wvRN-RL v *NetKit-CDI*	**0.400**	**0.100**	**0.050**	**0.025**	**0.025**	**0.025**	**0.025**	**0.025**	**0.050**
NetKit-CDI v *NB-local*	*0.400*	*0.400*	—	**0.450**	**0.450**	**0.250**	**0.400**	**0.200**	**0.200**

Table 21: *p*-values for the Wilcoxon signed-rank test comparing the performances of wvRN-RL (relaxation labeling), the method of Chakrabarti et al. (NetKit-CDI), and a local-only naive Bayes classifier. For each cell, bold text means that the first method was better than the second method and italics means it was worse. The cell with '—' means there was no significant difference.

although not significantly so in this comparison.[26] Most importantly, there are clear cases where using local-only methods as the sole baseline would lead to overly optimistic conclusions about the value of the full-blown relational learning technique.

This has an interesting implication: for classification purposes, as more labels are available in these networks, the network structure carries more information than what can be gotten from the local text, at least as measured using naive Bayes. It is important to reiterate that for different applications, the amount of local information and its relevance to the classification task can differ greatly, so these results are suggestive at best. Furthermore, aggregate classification accuracies tell only part of the story. Different subpopulations may be more or less amenable to local- or network-only classification (cf., fraud detection, Fawcett and Provost, 1997).

COMPARISON TO A GRAPH-BASED METHOD

Since there is a close relationship between wvRN and the "graph-based" methods that use only labels (as discussed in Sections 3.5.1 and 3.5.2), a question that follows from wvRN's excellent performance is whether such a graph-based method might provide an even better baseline than wvRN-RL. Blum et al. (2004) found that their randomized mincut method empirically did not perform as well as the method introduced by Zhu et al. (2003), so we compare the method of Zhu et al. to wvRN-RL on all the data from the case study. Their performances are nearly identical. This should come as no surprise as the two methods are nearly identical. Which method then should one use? By our experiments, the method of Zhu et al. gives slightly better scores for ranking entities by likelihood of class membership. Therefore, when their class mass normalization technique was employed to match the class posteriors to the class priors, Zhu's method generally improved more than when using class mass normalization with wvRN, making Zhu's method slightly but statistically significantly better than wvRN at $0.3 \leq r \leq 0.8$. However, one important reason to favor wvRN over the method of Zhu et al. is that the latter does not scale well: it requires inverting an NxN matrix and then doing a matrix multiplication, an O(N^3) operation. wvRN-RL is linear in the number of edges. In the worst case this is O(N^2), although most networks we have examined have a small average degree per node. For example, whereas for a modest-sized network (e.g., cora, N=4000) on a state-of-the-art compute server, wvRN finishes in just a few seconds, the method of Zhu et al.

26. The multi-data-set comparison has only 10 data points for the computation of statistical significance. The graphs also show standard-deviation error bars based on the 10 runs for each method, which suggest data-set specific conclusions of superiority. For example, in cora there is a clear ranking of the three methods.

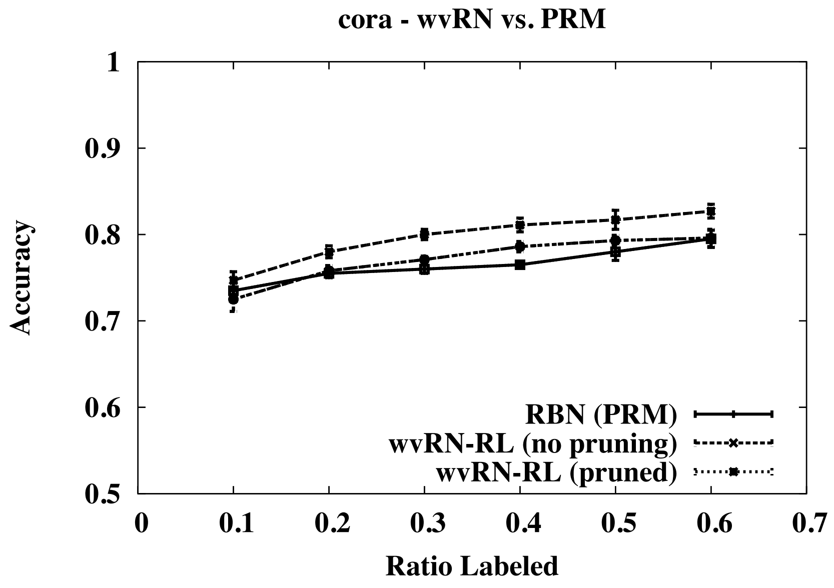


Figure 7: Comparison on cora of wvRN-RL (relaxation labeling) to RBN (PRM, Taskar et al., 2001). The graph shows wvRN using both citation and author links as in the original study. The "pruned" results follow the methodology of the case study in this paper by removing zero-knowledge components (see page 957) and singletons from the test set.

takes about 20 minutes. While this may not be a considerable burden for modest-sized data, some social network data sets are significantly larger (e.g., consider the many millions of communications records mined for fraud detection Fawcett and Provost, 1997; Cortes et al., 2001; Hill et al., 2006b or for network-based marketing Hill et al., 2006a, or by the NSA for counterterrorism, Tumulty, 2006). On a state-of-the-art machine, wvRN-RL can generate predictions on graphs consisting of a million nodes and millions of edges in minutes, whereas the method of Zhu et al. would not be tractable.

COMPARISON TO PRIOR PUBLISHED RESULTS

Yet another way to support the claim that network-only classification should be considered as a baseline is to a find published study for which using network-only classification would have changed the resulting conclusions. Consider cora. In a prior study, Taskar et al. (2001) show that a relational Bayesian network (RBN), there called a Probabilistic Relational Model (PRM), using author and citation links in conjunction with the text of the paper, was able to achieve a higher accuracy than a non-relational naive Bayesian classifier for $r = \{0.1, \ldots, 0.6\}$ (which did not use the links). However, as we saw above, wvRN performed quite well on this data set.

Figure 7 compares the accuracies of the RBN (transcribed from the graphs in the paper) with wvRN-RL. Note that in the prior work, they used fewer values of r and only had 5 cross-validation runs. Other than that, our setup is identical to the original study. Also note that to make a direct comparison, we here use both author edges and citation edges as was done in the original study (as opposed to only citation edges in our main study). We see clearly that wvRN was able to perform

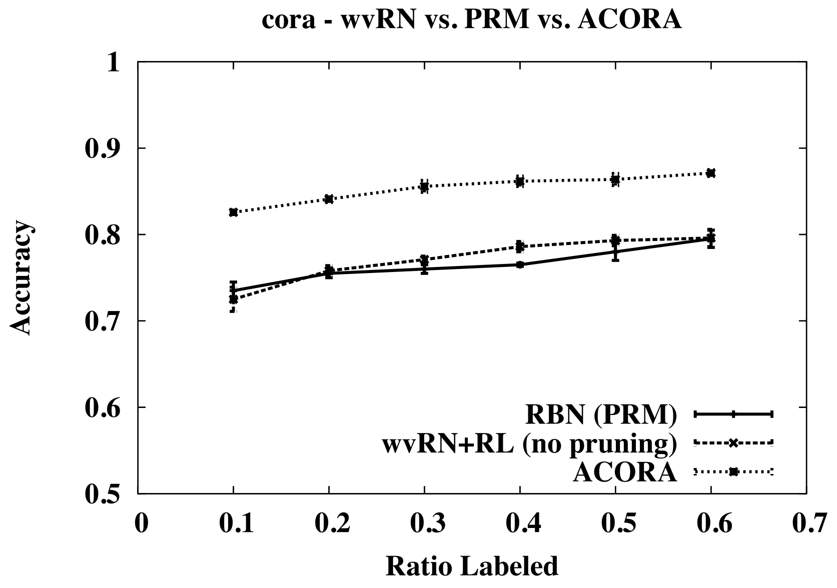


Figure 8: Comparison on cora of wvRN-RL (relaxation labeling) (no pruning, see Figure 7) to RBN (PRM, Taskar et al., 2001) and the use of identifier attributes via ACORA (Perlich and Provost, 2006).

comparably.[27] This demonstrates that cora is not a good data set to showcase the advantages of RBNs for classification. Had a method such as wvRN been readily available as a baseline, then Taskar et al. would most likely have used a more appropriate data set.

5.4 Limitations

We would like to characterize how much classification ability comes from the structure of the network alone. We have examined a limited notion: these methods all assume that "the structure of the network" can be reduced to "the structure of the neighborhood," bolstered by collective inference, rather than using relational models that look deeper.

Furthermore, we only considered links and class labels as comprising the structure of the network. We did not consider aspects of network structure such as the positions of individual nodes (as discussed in Section 3.5.3 on using node identifiers). For example, Figure 8 shows that accuracy can be improved substantially by reasoning with node identifiers via the ACORA system (Perlich and Provost, 2006). To our knowledge, no existing network classification system combines local attributes, node identifiers, and collective inference.

In the homogeneous, univariate case study we ignored much of the complexity of real networked data, such as heterogeneous edges, heterogeneous nodes, directed edges, and attributes of nodes and edges. Each of these introduces complications and opportunities for modeling. There are no

27. The "pruned" results show the accuracy after eliminating the zero-knowledge components, for which wvRN-RL can only predict the most prevalent class. Recall that zero-knowledge components (as defined on page 957) are nodes for which there is no path to any node in $\mathbf{V}^K$.

comprehensive, empirical machine learning studies that consider these dimensions systematically. For example, when using attributes of nodes, how much is gained by using them in the relational classifier, as opposed to using them simply to initialize priors? (For example, Chakrabarti et al. 1998 found that using the text of hyperlinked documents reduced performance.) Similarly, how much value is added by considering multiple edge types explicitly?

An important limitation of this work, with respect to its relevance to practical problems, is that we chose training data randomly to be labeled. It is likely that the data for which labels are available are interdependent. For example, all the members from one terrorist cell may be known and none from another. If other attributes are available more uniformly, then studies such as this may artificially favor network-only methods over attribute-based methods.

5.5 Conclusions and Future Work

We introduced a modular toolkit, NetKit-SRL, for classification in networked data. The importance of NetKit is three-fold: (1) it generalizes several existing methods for classification in networked data, thereby making comparison to existing methods possible; (2) it enables the creation and use of many new algorithms by its modularity and extensibility, and (3) it enables the analysis/comparison of individual components and configurations.

We then used NetKit to perform a case study of within-network, univariate classification for homogeneous networked data. The case study makes several contributions. It provides demonstrative support for points 2 and 3 above. Of the five network classifiers that stood clearly above the rest, three were novel combinations of components (and thus, novel relational classifiers). Certain collective inference and relational classification components stood out with consistently better performance. For collective inference, relaxation labeling was best when there are few known labels; it mattered little which collective inference method is used when there were many known labels. For relational classification, the link-based classifier clearly was preferable when many labels were known. The lower-variance methods (wvRN and cdRN) dominated when fewer labels were known. In combination, two pairs of relational classifier/collective inference combinations stand out strikingly: nLB with any of the collective inference methods dominates when many labels are known; the relational neighbor methods with relaxation labeling (wvRN-RL and cdRN-RL) dominate when fewer labels are known. Of particular relevance given this latter result, we also show the close correspondence of several simple, network classification methods: the node-centric wvRN-RL (Macskassy and Provost, 2003), the random-field method of Zhu et al. (2003), and classic connectionist techniques such as Hopfield networks (Hopfield, 1982).

The case study demonstrates the power of simple network classification models, and argues that they must be included when evaluating relational learning methods on networked data. On the benchmark data sets, error rates were reduced substantially by taking into account only the class-linkage structure of the network. Although learning helped in many cases, the no-learning wvRN was a very strong competitor—performing very well especially when few labels were known. This result was supported further by followup analyses showing cases where one would draw overly optimistic conclusions about the value of more-sophisticated methods if these network-only techniques were not used as baselines. It also raises the question of whether we need "better" benchmark data sets—to showcase the value of more-powerful techniques.

More generally, the results showcase two different modes of within-network classification that call for different types of methods: when many labels are known versus when few labels are known.

These modes correspond to different application scenarios. The former may correspond (for example) to networks that evolve over time with new nodes needing classification before their labels become known naturally, as would be the case for predicting movie box-office receipts. Examples of the little-known scenario can be found in counter-terrorism and law enforcement, where analysts form complex interaction networks containing a few, known bad guys. The little-known scenario has an economic component, similar to active learning: it may be worthwhile to incur costs to label additional nodes in the network, because this will lead to much improved classification. This suggests another direction for future work—identifying the most beneficial nodes for labeling (cf., Domingos and Richardson, 2001).

The case study also showcases a problem of representation for network classification: the selection of which edges to use. We show that edge selection can make a considerable difference, and suggest techniques analogous to those used in traditional feature selection. It is straightforward to extend NetKit's relational classification methods to handle heterogeneous links. However, that would not solve the fundamental problem that edge selection (like feature selection) may improve generalization performance, as well as provide simpler models.

Classification in networked data is important for real-world applications and presents many opportunities for machine-learning research. The field is beginning to amass benchmark domains containing networked data. We hope that NetKit can facilitate systematic study.

Acknowledgments

David Jensen made many helpful suggestions, including pointing us to the WebKB data set and suggesting ways to do well on it. Abraham Bernstein collaborated on the generation of the two industry classification data sets. Kaveh R. Ghazi worked diligently coding the open-source NetKit and ensuring that it would be ready for release on schedule. We thank Ben Taskar and Andrew McCallum for providing us with versions of the cora data set. Thanks to Sugato Basu, Misha Bilenko, Avrim Blum, Pedro Domingos, Joydeep Ghosh, Michael Kearns, Yann LeCun, Andrew McCallum, Jennifer Neville, Mark Newman, Claudia Perlich, and audience members of talks for helpful comments and discussions. Finally, we thank Andrew McCallum and the anonymous reviewers who helped to strengthen the paper considerably.

This work was funded in part by a grant from the New York Software Industry Association. This research also was sponsored in part by the Air Force Research Laboratory, Air Force Materiel Command, USAF, under Agreement number F30602-01-2-0585. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFRL or the U.S. Government.

References

J. C. Almack. The influence of intelligence on the selection of associates. *School and Society*, 16: 529–530, 1922.

A. Bernstein, S. Clearwater, and F. Provost. The relational vector-space model and industry classification. In *Proceedings of the Learning Statistical Models from Relational Data Workshop at the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society*, 36(2):192–236, 1974.

J. Besag. Statistical analysis of non-lattice data. *The Statistician*, 24(3):179–195, 1975.

J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society*, 48(3): 259–302, 1986.

P. M. Blau. *Inequality and Heterogeneity: A Primitive Theory of Social Structure*. New York: Free Press, 1977.

A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, pages 19–26, 2001.

A. Blum, J. Lafferty, R. Reddy, and M. R. Rwebangira. Semi-supervised learning using randomized mincuts. In *Proceedings of the Twentyfirst International Conference on Machine Learning (ICML)*, pages 97–104, 2004.

H. Bott. Observation of play activities in a nursery school. *Genetic Psychology Monographs*, 4: 44–88, 1928.

Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(11):1222–1239, 2001.

S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 307–319, 1998.

C. Cortes, D. Pregibon, and C. T. Volinsky. Communities of interest. In *Proceedings of the Fourth International Conference on Advances in Intelligent Data Analysis (IDA)*, pages 105–114, 2001.

R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic networks and expert systems*. Springer, 1999.

M. Craven, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and C. Y. Quek. Learning to extract symbolic knowledge from the World Wide Web. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI)*, pages 509–516, 1998.

L. De Raedt, H. Blockeel, L. Dehaspe, and W. Van Laer. Three companions for data mining in first order logic. In S. Dzeroski and N. Lavrac, editors, *Relational Data Mining*, pages 105–139. Berlin; New York: Springer, 2001.

I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 269–274, 2001.

R. L. Dobrushin. The description of a random field by means of conditional probabilities and conditions of its regularity. *Theory of Probability and its Applications*, 13(2):197–224, 1968.

P. Domingos and M. Richardson. Mining the network value of customers. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 57–66, 2001.

S. Dzeroski and N. Lavrac. *Relational Data Mining*. Berlin; New York: Springer, 2001.

T. Fawcett and F. Provost. Adaptive fraud detection. *Data Mining and Knowledge Discovery*, 3: 291–316, 1997.

T. Fawcett and F. Provost. Activity monitoring: Noticing interesting changes in behavior. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 53–62, 1999.

P. A. Flach and N. Lachiche. Naive Bayesian classification of structured data. *Machine Learning*, 57:233-269, 2004.

N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1300–1309, 1999.

A. Galstyan and P. Cohen. Is guilt by association a bad thing? In *Proceedings of the First International Conference on Intelligence Analysis (IA)*, 2005.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 6:721–741, 1984.

W. R. Gilks, S. Richardson, and D. J. Spiegelhalter. *Markov Chain Monte Carlo in Practice*. Chapman & Hall/CRC, 1995.

D. Greig, B. Porteous, and A. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society*, 51(2):271–279, 1989.

D. D. Heckathorn and J. Jeffri. Jazz networks: Using respondent-driven sampling to study stratification in two jazz musician communities. Unpublished paper presented at American Sociological Association Annual Meeting, August 2003.

D. Heckerman, D. M. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research (JMLR)*, 1:49–75, 2000.

S. Hill, F. Provost, and C. Volinsky. Network-based marketing: Identifying likely adopters via consumer networks. *Statistical Science*, 22(2):256–276, 2006a.

S. Hill, D.K. Agarwal, R. Bell, and C. Volinsky. Building an effective representation for dynamic networks. *Journal of Computational & Graphical Statistics*, 15(3):584–608, 2006b.

G. E. Hinton and T. J. Sejnowski. Learning and relearning in Boltzmann machines. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, 1: Foundations:282–317, 1986.

J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79(8): 2554–2558, 1982.

Z. Huang, H. Chen, and D. Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems (TOIS)*, 22(1): 116–142, 2004.

R. A. Hummel and S. W. Zucker. On the foundations of relaxation labeling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 5(3):267–287, 1983.

E. Ising. Beitrag zur Theorie des Ferromagnetismus. *Zeitschrift f. Physik*, 31:253–258, 1925. [German].

D. Jensen and J. Neville. Data mining in social networks. In *National Academy of Sciences Symposium on Dynamic Social Network Modeling and Analysis*, 2002a.

D. Jensen and J. Neville. Linkage and autocorrelation cause feature selection bias in relational learning. In *Proceedings of the Nineteenth International Conference on Machine Learning (ICML)*, pages 259–266, 2002b.

D. Jensen, J. Neville, and B. Gallagher. Why collective inference improves relational classification. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 593–598, 2004.

T. Joachims. Transductive learning via spectral graph partitioning. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, pages 290–297, 2003.

J. Kleinberg and É. Tardos. Approximation algorithms for classification problems with pairwise relations: Metric labeling and Markov random fields. In *Proceedings of the Fortieth Annual Symposium on Foundations of Computer Science (FOCS)*, pages 14–23, 1999.

R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1–2): 273–324, 1997.

D. Koller and A. Pfeffer. Probabilistic frame-based systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI)*, pages 580–587, 1998.

S. Kramer, N. Lavrac, and P. Flach. Propositionalization approaches to relational data mining. In S. Dzeroski and N. Lavrac, editors, *Relational Data Mining*, pages 262–291. Berlin; New York: Springer, 2001.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, pages 282–289, 2001.

P. Langley. Crafting papers on machine learning. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML)*, pages 1207–1212, 2000.

P. Lazarsfeld and R. K. Merton. Friendship as a social process: A substantive and methodological analysis. In M. Berger, T. Abel, and C. H. Page, editors, *Freedom and Control in Modern Society*, pages 18–66. Van Nostrand, 1954.

C. P. Loomis. Political and occupational cleavages in a Hanoverian village. *Sociometry*, 9:316–3333, 1946.

Q. Lu and L. Getoor. Link-based classification. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, pages 496–503, 2003.

S. A. Macskassy and F. Provost. A simple relational classifier. In *Proceedings of the Multi-Relational Data Mining Workshop (MRDM) at the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003.

S. A. Macskassy and F. Provost. Suspicion scoring based on guilt-by-association, collective inference, and focused data access. In *Proceedings of the First International Conference on Intelligence Analysis (IA)*, 2005.

A. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

M. McPherson, L. Smith-Lovin, and J. M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27:415–444, 2001.

K. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief-propagation for approximate inference: An empirical study. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 467–475, 1999.

J. Neville and D. Jensen. Iterative classification in relational data. In *Proceedings of the Workshop on Learning Statistical Models from Relational Data at the Seventeenth National Conference on Artificial Intelligence (AAAI)*, pages 13–20, 2000.

J. Neville and D. Jensen. Collective classification with relational dependency networks. In *Proceedings of the Multi-Relational Data Mining Workshop (MRDM) at the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003.

J. Neville and D. Jensen. Dependency networks for relational data. In *Proceedings of the Fourth IEEE International Conference in Data Mining (ICDM)*, pages 170–177, 2004.

J. Neville and D. Jensen. Leveraging relational autocorrelation with latent group models. In *Proceedings of the Fifth IEEE International Conference in Data Mining (ICDM)*, pages 322–329, 2005.

J. Neville and D. Jensen. Relational dependency networks. *Journal of Machine Learning Research (JMLR)*, 8(Mar):653–692, 2007.

J. Neville, D. Jensen, L. Friedland, and M. Hay. Learning relational probability trees. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 625–630, 2003.

J. Neville, Ö. Şimşek, D. Jensen, J. Komoroske, K. Palmer, and H. Goldberg. Using relational knowledge discovery to prevent securities fraud. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 449–458, 2005.

M. E. J. Newman. Mixing patterns in networks. *Physical Review E*, 67, 2003. 026126.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

C. Perlich. Citation-based document classification. In *Workshop on Information Technology and Systems (WITS)*, 2003.

C. Perlich and F. Provost. Aggregation-based feature invention and relational concept classes. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 167–176, 2003.

C. Perlich and F. Provost. Distribution-based aggregation for relational learning with identifier attributes. *Machine Learning*, 62(1/2):65–105, 2006.

C. Perlich, F. Provost, and J. Simonoff. Tree induction vs. logistic regression: A learning-curve analysis. *Journal of Machine Learning Research (JMLR)*, 4:211-255, 2003.

A. Popescul and L. H. Ungar. Statistical relational learning for link prediction. In *Proceedings of the Learning Statistical Models from Relational Data Workshop at the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

R. B. Potts. Some generalized order-disorder transformations. *Cambridge Philosophic Society*, 48: 106–109, 1952.

H. M. Richardson. Community of values as a factor in friendships of college and adult women. *Journal of Social Psychology*, 11:303–312, 1940.

M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1/2):107–136, 2006.

J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System: Experiments in Automatic Document Processing*, chapter 14, pages 313–323. Prentice–Hall, 1971.

A. Rosenfeld, R. Hummel, and S. Zucker. Scene labeling by relaxation operations. *IEEE Transactions on Systems, Man and Cybernetics*, 6:420–433, 1976.

L. J. Savage. *The Foundations of Statistics*. John Wiley and Sons, 1954.

E. Segal, H. Wang, and D. Koller. Discovering molecular pathways from protein interaction and gene expression data. *Bioinformatics*, 19:I264–I272, Jul 2003a.

E. Segal, R. Yelensky, and D. Koller. Genome-wide discovery of transcriptional modules from DNA sequence and gene expression. *Bioinformatics*, 19:I273–I282, Jul 2003b.

B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 485–492, 2002.

B. Taskar, E. Segal, and D. Koller. Probabilistic classification and clustering in relational data. In *Proceedings of the Seventeenthth International Joint Conference on Artificial Intelligence (IJ-CAI)*, pages 870–878, 2001.

B. Taskar, V. Chatalbashev, and D. Koller. Learning associative Markov networks. In *Proceedings of the Twentyfirst International Conference on Machine Learning (ICML)*, 2004.

K. Tumulty. Inside Bush's Secret Spy Net [electronic version]. *Time*, 167(21), 2006. Retrieved May 25, 2006, from `http://www.time.com/time/archive/preview/0,10987,1194021,00.html`.

V. N. Vapnik. The support vector method of function estimation. In J. Suykens and J. Vandewalle, editors, *Nonlinear Modeling: Advanced Black-Box Techniques*, pages 55–86. Kluwer, Boston, 1998a.

V. N. Vapnik. *Statistical Learning Theory*. John Wiley, NY, 1998b.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. Technical Report 649, University of California, Berkeley, 2003.

G. Winkler. *Image Analysis, Random Fields and Markov Chain Monte Carlo Methods*. Springer-Verlag, 2nd edition, 2003.

I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools with Java Implementations*. Morgan Kaufmann, San Francisco, 2000.

X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, pages 912–919, 2003.

Journal of Machine Learning Research 8 (2007) 985-1005 Submitted 6/06; Revised 8/06; Published 5/07

Covariate Shift Adaptation by Importance Weighted Cross Validation

Masashi Sugiyama SUGI@CS.TITECH.AC.JP
Department of Computer Science
Tokyo Institute of Technology
2-12-1, O-okayama, Meguro-ku, Tokyo, 152-8552, Japan

Matthias Krauledat MATTHIAS.KRAULEDAT@FIRST.FHG.DE
Klaus-Robert Müller KLAUS@FIRST.FHG.DE
Department of Computer Science
Technical University Berlin
Franklinstr. 28/29, 10587 Berlin, Germany

Editor: Yoshua Bengio

Abstract

A common assumption in supervised learning is that the input points in the training set follow the *same* probability distribution as the input points that will be given in the future test phase. However, this assumption is not satisfied, for example, when the outside of the training region is extrapolated. The situation where the training input points and test input points follow *different* distributions while the conditional distribution of output values given input points is unchanged is called the *covariate shift*. Under the covariate shift, standard model selection techniques such as cross validation do not work as desired since its unbiasedness is no longer maintained. In this paper, we propose a new method called *importance weighted cross validation* (IWCV), for which we prove its unbiasedness even under the covariate shift. The IWCV procedure is the only one that can be applied for unbiased classification under covariate shift, whereas alternatives to IWCV exist for regression. The usefulness of our proposed method is illustrated by simulations, and furthermore demonstrated in the brain-computer interface, where strong non-stationarity effects can be seen between training and test sessions.

Keywords: covariate shift, cross validation, importance sampling, extrapolation, brain-computer interface

1. Introduction

The goal of supervised learning is to infer an unknown input-output dependency from training samples, by which output values for unseen test input points can be estimated. When developing a method of supervised learning, it is commonly assumed that the input points in the training set and the input points used for testing follow the *same* probability distribution (e.g., Wahba, 1990; Bishop, 1995; Vapnik, 1998; Duda et al., 2001; Hastie et al., 2001; Schölkopf and Smola, 2002). However, this common assumption is not fulfilled, for example, when we extrapolate outside of the training region[1] or when training input points are designed by an active learning (experimental design) algorithm. The situation where the training input points and test input points follow different

1. The term 'extrapolation' could have been defined in a narrow sense as prediction in regions with *no* training samples. On the other hand, the situation we are considering here is 'weak' extrapolation; prediction is carried out in the region where only a small number of training samples is available.

probability distributions but the conditional distributions of output values given input points are unchanged is called the *covariate shift* (Shimodaira, 2000). For data from many applications such as off-policy reinforcement learning (Shelton, 2001), spam filtering (Bickel and Scheffer, 2007), bioinformatics (Baldi et al., 1998; Borgwardt et al., 2006) or brain-computer interfacing (Wolpaw et al., 2002), the covariate shift phenomenon is conceivable. Sample selection bias (Heckman, 1979) in economics may also include a form of the covariate shift. Illustrative examples of covariate shift situations are depicted in Figures 1 and 3.

In this paper, we develop a new learning method and prove that we can alleviate misestimation due to covariate shift. From the beginning, we note that all the theoretical discussions will be made under the assumption that the *ratio* of test and training input densities at training input points is known; in experimental studies, the density ratio will be replaced by their empirical estimates and the practical performance of our approach will be evaluated.

Model selection is one of the key ingredients in machine learning. However, under the covariate shift, a standard model selection technique such as *cross validation* (CV) (Stone, 1974; Wahba, 1990) does not work as desired; more specifically, the unbiasedness that guarantees the accuracy of CV does not hold under the covariate shift anymore. To cope with this problem, we propose a novel variant of CV called *importance weighted CV* (IWCV). We prove that IWCV gives an almost unbiased estimate of the risk even under the covariate shift. Model selection under the covariate shift has been studied so far only by few researchers (e.g., Shimodaira, 2000; Sugiyama and Müller, 2005)—existing methods have a number of limitations, for example, in the loss function, parameter learning method, and model. In particular, the existing methods can not be applied to classification scenarios. On the other hand, the proposed IWCV overcomes these limitations: it allows for *any* loss function, parameter learning method, and model; even non-parametric learning methods can be employed. To the best of our knowledge, the proposed IWCV is the first method that can be successfully applied to model selection in covariate-shifted classification tasks. The usefulness of the proposed method is demonstrated in the brain-computer interface applications, in which existing methods for covariate shift compensation could not be employed.

2. Problem Formulation

In this section, we formulate the supervised learning problem with the covariate shift, and review existing learning methods.

2.1 Supervised Learning under Covariate Shift

Let us consider the supervised learning problem of estimating an unknown input-output dependency from training samples. Let $\mathcal{T} = \{(x_i, y_i)\}_{i=1}^{n}$ be the training samples, where $x_i \in \mathcal{X} \subset \mathbb{R}^d$ is an i.i.d. training input point following a probability distribution $P_{train}(x)$ and $y_i \in \mathcal{Y} \subset \mathbb{R}$ is a corresponding training output value following a conditional probability distribution $P(y|x)$. $P(y|x)$ may be regarded as the sum of true output $f(x)$ and noise.

Let $\ell(x, y, \widehat{y}) : \mathcal{X} \times \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$ be the loss function, which measures the discrepancy between the true output value y at an input point x and its estimate $\widehat{y}$. Let us employ a parametric model $\widehat{f}(x; \theta)$ for estimating the output value y, where $\theta \in \Theta \subset \mathbb{R}^b$. Note that the range of application of our proposed method given in Section 3 includes non-parametric methods, but we focus on a parametric setting for simplicity. A model $\widehat{f}(x; \theta)$ is said to be *correctly specified* if there exists a parameter θ^* such that $\widehat{f}(x; \theta^*) = f(x)$; otherwise the model is said to be *misspecified*. In practice, the model used for learning would be misspecified to a greater or lesser extent. For this reason, we do not assume that the model is correct in this paper. The goal of supervised learning is to determine the value of the parameter θ so that output values for unlearned test input points are accurately estimated.

Let us consider a test sample, which is not given to the user in the training phase, but will be given in the test phase in the future. We denote the test sample by (t,u), where $t \in \mathcal{X}$ is a test input point and $u \in \mathcal{Y}$ is a corresponding test output value. The test error expected over test samples is expressed as

$$\mathbb{E}_{t,u}\left[\ell(t,u,\widehat{f}(t;\widehat{\theta}))\right], \tag{1}$$

where $\mathbb{E}$ denotes the expectation. Note that the learned parameter $\widehat{\theta}$ generally depends on the training set $\mathcal{T} = \{(x_i,y_i)\}_{i=1}^{n}$. In the following, we consider the expected test error over the training samples, which is called the *risk* or the *generalization error*:

$$R^{(n)} \equiv \mathbb{E}_{\{x_i,y_i\}_{i=1}^{n},t,u}\left[\ell(t,u,\widehat{f}(t;\widehat{\theta}))\right]. \tag{2}$$

In standard supervised learning theories, the test sample (t,u) is assumed to follow $P(u|t)P_{train}(t)$, which is the *same* probability distribution as for the training samples $\{(x_i,y_i)\}_{i=1}^{n}$ (e.g., Wahba, 1990; Bishop, 1995; Vapnik, 1998; Duda et al., 2001; Hastie et al., 2001; Schölkopf and Smola, 2002). On the other hand, in this paper, we consider the situation under the *covariate shift*, that is, the conditional distribution $P(u|t)$ remains unchanged, but the test input point t follows a different probability distribution $P_{test}(x)$. Illustrative examples of covariate shift situations are depicted in Figures 1 and 3.

Let $p_{train}(x)$ and $p_{test}(x)$ be the probability density functions corresponding to the input distributions $P_{train}(x)$ and $P_{test}(x)$, respectively. In the following theoretical discussions, we assume that the *ratio* of test and training input densities at training input points,

$$\frac{p_{test}(x_i)}{p_{train}(x_i)}, \tag{3}$$

is finite and known. We refer to the expression (3) as *importance* à la *importance sampling* (Fishman, 1996). In practical situations where the importance is unknown, we may replace them by empirical estimates (see Sections 4 and 5).

2.2 Empirical Risk Minimization and Its Importance Weighted Variants

A standard method to learn the parameter θ would be *empirical risk minimization* (ERM) (e.g., Vapnik, 1998; Schölkopf and Smola, 2002):

$$\widehat{\theta}_{ERM} \equiv \underset{\theta\in\Theta}{\operatorname{argmin}}\left[\frac{1}{n}\sum_{i=1}^{n}\ell(x_i,y_i,\widehat{f}(x_i;\theta))\right].$$

If $P_{train}(x) = P_{test}(x)$, ERM provides a *consistent* estimator[2] (Shimodaira, 2000). However, under the covariate shift where $P_{train}(x) \neq P_{test}(x)$, ERM is not generally consistent anymore[3] (Shimodaira,

2. For a correctly specified model, an estimator is said to be consistent if it converges in probability to the *true* parameter. For a misspecified model, we say that an estimator is consistent if it converges in probability to the *optimal* parameter in the model (i.e., the optimal approximation of the learning target under the risk (2) within the model).

3. If the model is correct, ERM is still consistent even under the covariate shift.

2000):

$$\lim_{n\to\infty}\left(\widehat{\theta}_{ERM}\right) \neq \theta^*,$$

where

$$\theta^* \equiv \operatorname*{argmin}_{\theta\in\Theta}\left(\mathbb{E}_{t,u}\left[\ell(t,u,\widehat{f}(t;\theta))\right]\right).$$

Under the covariate shift, the following *importance weighted ERM* (IWERM) is consistent (Shimodaira, 2000):

$$\widehat{\theta}_{IWERM} \equiv \operatorname*{argmin}_{\theta\in\Theta}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{p_{test}(x_i)}{p_{train}(x_i)}\ell(x_i,y_i,\widehat{f}(x_i;\theta))\right],$$

which satisfies even for a misspecified model

$$\lim_{n\to\infty}\left(\widehat{\theta}_{IWERM}\right) = \theta^*.$$

Although IWERM is consistent, it is not *efficient* and can be rather unstable (Shimodaira, 2000). Therefore, IWERM may not be optimal in practical finite sample cases; a slightly stabilized variant of IWERM could be practically better than plain IWERM. The trade-off between consistency and stability could be controlled, for example, by weakening the weight (*Adaptive IWERM*; AIWERM) or by adding a regularizer to the empirical risk (*Regularized IWERM*; RIWERM):

$$\widehat{\theta}_{AIWERM} \equiv \operatorname*{argmin}_{\theta\in\Theta}\left[\frac{1}{n}\sum_{i=1}^{n}\left(\frac{p_{test}(x_i)}{p_{train}(x_i)}\right)^{\lambda}\ell(x_i,y_i,\widehat{f}(x_i;\theta))\right], \tag{4}$$

$$\widehat{\theta}_{RIWERM} \equiv \operatorname*{argmin}_{\theta\in\Theta}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{p_{test}(x_i)}{p_{train}(x_i)}\ell(x_i,y_i,\widehat{f}(x_i;\theta))+\gamma R(\theta)\right],$$

where $0 \leq \lambda \leq 1, \gamma \geq 0$, and $R(\theta)$ is some regularization functional.

The above AIWERM and RIWERM methods are just examples; there may be many other possibilities of controlling the trade-off between consistency and stability. We note that the methodology we propose in this paper is valid for *any* parameter learning method; this means that, e.g., an importance weighted variant of support vector machines (Vapnik, 1998; Schölkopf and Smola, 2002; Huang et al., 2007) or graph regularization techniques (Bousquet et al., 2004; Belkin and Niyogi, 2004; Hein, 2006) can also be employed.

2.3 Cross Validation Estimate of Risk

The value of the tuning parameter, say λ in Eq. (4), controls the trade-off between the consistency and stability. Therefore, we need to perform *model selection* for determining the value of λ. *Cross validation* (CV) is a popular method for model selection (Stone, 1974; Wahba, 1990). A basic idea of CV is to divide the training set into 'training part' and 'validation part'—a learning machine is trained using the training part and is tested using the validation part, by which the risk is estimated. Below, we give a slightly more formal description of the CV procedure, which will be used in the next section.

Let us randomly divide the training set $\mathcal{T} = \{(x_i,y_i)\}_{i=1}^{n}$ into k disjoint non-empty subsets $\{\mathcal{T}_i\}_{i=1}^{k}$. Let $\widehat{f}_{\mathcal{T}_j}(x)$ be a function learned from $\{\mathcal{T}_i\}_{i\neq j}$. Then the *k-fold CV* (*k*CV) estimate of the risk $R^{(n)}$ is given by

$$\widehat{R}^{(n)}_{kCV} \equiv \frac{1}{k}\sum_{j=1}^{k}\frac{1}{|\mathcal{T}_j|}\sum_{(x,y)\in\mathcal{T}_j}\ell(x,y,\widehat{f}_{\mathcal{T}_j}(x)),$$

where $|\mathcal{T}_j|$ is the number of samples in the subset $\mathcal{T}_j$. When $k=n$, kCV is particularly called *leave-one-out cross validation* (LOOCV).

$$\widehat{R}^{(n)}_{LOOCV} \equiv \frac{1}{n}\sum_{j=1}^{n}\ell(x_j,y_j,\widehat{f}_j(x_j)),$$

where $\widehat{f}_j(x)$ is a function learned from $\{(x_i,y_i)\}_{i\neq j}$.

It is known that, if $P_{train}(x)=P_{test}(x)$, LOOCV gives an *almost* unbiased estimate of the risk; more precisely, LOOCV gives an unbiased estimate of the risk with $n-1$ samples (Luntz and Brailovsky, 1969; Schölkopf and Smola, 2002):

$$\mathbb{E}_{\{x_i,y_i\}_{i=1}^{n}}\left[\widehat{R}^{(n)}_{LOOCV}\right]=R^{(n-1)}\approx R^{(n)}.$$

However, this useful property is no longer true under the covariate shift. In the following section, we give a novel modified cross validation method which still maintains the 'almost unbiasedness' property even under the covariate shift.

3. Importance Weighted Cross Validation

To compensate for the effect of the covariate shift in the CV procedure, we propose the following *importance weighted CV* (IWCV):

$$\widehat{R}^{(n)}_{kIWCV} \equiv \frac{1}{k}\sum_{j=1}^{k}\frac{1}{|\mathcal{T}_j|}\sum_{(x,y)\in\mathcal{T}_j}\frac{p_{test}(x)}{p_{train}(x)}\ell(x,y,\widehat{f}_{\mathcal{T}_j}(x)),$$

or

$$\widehat{R}^{(n)}_{LOOIWCV} \equiv \frac{1}{n}\sum_{j=1}^{n}\frac{p_{test}(x_j)}{p_{train}(x_j)}\ell(x_j,y_j,\widehat{f}_j(x_j)).$$

That is, the validation error in the CV procedure is weighted according to the importance.

The following lemma shows that LOOIWCV gives an almost unbiased estimate of the risk even under the covariate shift.

Lemma 1

$$\mathbb{E}_{\{x_i,y_i\}_{i=1}^{n}}\left[\widehat{R}^{(n)}_{LOOIWCV}\right]=R^{(n-1)}.$$

Proof Since the conditional distribution $P(y|x)$ does not change between the training and test phases, we have, for any $j \in \{1,\ldots,n\}$,

$$\begin{aligned}
&\mathbb{E}_{\{x_i,y_i\}_{i=1}^n}\left[\frac{p_{test}(x_j)}{p_{train}(x_j)}\ell(x_j,y_j,\widehat{f}_j(x_j))\right] \\
&\quad = \mathbb{E}_{\{x_i,y_i\}_{i\neq j},x_j,y_j}\left[\frac{p_{test}(x_j)}{p_{train}(x_j)}\ell(x_j,y_j,\widehat{f}_j(x_j))\right] \\
&\quad = \mathbb{E}_{\{x_i,y_i\}_{i\neq j},y_j}\left[\int_{\mathcal{X}}\frac{p_{test}(x_j)}{p_{train}(x_j)}\ell(x_j,y_j,\widehat{f}_j(x_j))p_{train}(x_j)dx_j\right] \\
&\quad = \mathbb{E}_{\{x_i,y_i\}_{i\neq j},y_j}\left[\int_{\mathcal{X}}p_{test}(x_j)\ell(x_j,y_j,\widehat{f}_j(x_j))dx_j\right] \\
&\quad = \mathbb{E}_{\{x_i,y_i\}_{i\neq j},u}\left[\int_{\mathcal{X}}\ell(t,u,\widehat{f}_j(t))p_{test}(t)dt\right] \\
&\quad = \mathbb{E}_{\{x_i,y_i\}_{i\neq j},t,u}\left[\ell(t,u,\widehat{f}_j(t))\right] \\
&\quad = R^{(n-1)}.
\end{aligned}$$

Then we have

$$\begin{aligned}
\mathbb{E}_{\{x_i,y_i\}_{i=1}^n}\left[\widehat{R}^{(n)}_{LOOIWCV}\right] &= \mathbb{E}_{\{x_i,y_i\}_{i=1}^n}\left[\frac{1}{n}\sum_{j=1}^n\frac{p_{test}(x_j)}{p_{train}(x_j)}\ell(x_j,y_j,\widehat{f}_j(x_j))\right] \\
&= \frac{1}{n}\sum_{j=1}^n\mathbb{E}_{\{x_i,y_i\}_{i=1}^n}\left[\frac{p_{test}(x_j)}{p_{train}(x_j)}\ell(x_j,y_j,\widehat{f}_j(x_j))\right] \\
&= \frac{1}{n}\sum_{j=1}^n R^{(n-1)} \\
&= R^{(n-1)},
\end{aligned}$$

which concludes the proof. ■

As proved above, the simple variant of LOOCV called LOOIWCV provides an unbiased estimate of the risk with $n-1$ samples even under the covariate shift. A similar proof is also possible for kIWCV, although its bias may be larger than LOOIWCV. Note that we did not assume any condition on the loss function in the above proof. This means that the almost unbiasedness is valid for any loss function including non-smooth losses such as the 0/1-loss. Also, we did not make a model-specific assumption in the proof. Therefore, the almost unbiasedness holds for any model; even non-identifiable models (Watanabe, 2001) such as multi-layer perceptrons or Gaussian mixture models are included. Furthermore, the above proof does not depend on the method of parameter learning. This means that the almost unbiasedness is valid for any parameter learning method; even non-parametric learning methods are allowed.

4. Numerical Examples

In this section, we illustrate how IWCV works using toy regression and classification data sets. More simulation results can be found in a separate technical report (Sugiyama et al., 2006).

4.1 Toy Regression Problem

Here we illustrate the behavior of the proposed and existing generalization error estimators using a simple one-dimensional regression data set.

We use the following linear model for learning:

$$\widehat{f}(x;\theta) = \theta_0 + \sum_{i=1}^{d} \theta_i x^{(i)}, \tag{5}$$

where $x^{(i)}$ is the ith element of x and d is the input dimensionality. The parameter vector θ is learned by *adaptive importance weighted least-squares* (AIWLS):

$$\widehat{\theta}_{AIWLS} \equiv \operatorname*{argmin}_{\theta} \left[\frac{1}{n} \sum_{i=1}^{n} \left(\frac{p_{test}(x_i)}{p_{train}(x_i)} \right)^{\lambda} \left(\widehat{f}(x_i;\theta) - y_i \right)^2 \right], \tag{6}$$

where $0 \leq \lambda \leq 1$; λ is chosen later by a model selection method. For the linear model (5), the above minimizer $\widehat{\theta}_{AIWLS}$ is given analytically by

$$\widehat{\theta}_{AIWLS} = (X^\top D^\lambda X)^{-1} X^\top D^\lambda y,$$

where

$$X \equiv \begin{pmatrix} 1 & x_1^\top \\ 1 & x_2^\top \\ & \vdots \\ 1 & x_n^\top \end{pmatrix},$$

which is assumed to have rank $d+1$; D is the diagonal matrix with the ith diagonal element $D_{i,i} = p_{test}(x_i)/p_{train}(x_i)$ and $y = (y_1, y_2, \ldots, y_n)^\top$.

Let the learning target function be $f(x) = \mathrm{sinc}(x)$ and let the training and test input densities be

$$p_{train}(x) = \phi(x; 1, (1/2)^2),$$
$$p_{test}(x) = \phi(x; 2, (1/4)^2),$$

where $\phi(x;\mu,\sigma^2)$ is the normal density with mean μ and variance σ^2. This setting implies that we are considering an extrapolation problem (see Figure 1(A)). We create the training output value y_i ($i = 1, 2, \ldots, n$) as $y_i = f(x_i) + \varepsilon_i$, where $\{\varepsilon_i\}_{i=1}^n$ have density $\phi(\varepsilon; 0, (1/4)^2)$. Let the number of training samples be $n = 150$.

Figure 1 (B)–(D) illustrate the true function, a realization of training samples, learned functions by AIWLS (6) with $\lambda = 0, 0.5, 1$, and a realization of test samples. For this particular realization, $\lambda = 0.5$ appears to work very well. However, the best choice of λ depends on the realization of samples and λ needs to be carefully chosen by a model selection method. We randomly create $\{x_i, \varepsilon_i\}_{i=1}^n$ and calculate the scores of 10-fold IWCV and 10-fold CV for $\lambda = 0, 0.1, 0.2, \ldots, 1$. This procedure is repeated 1000 times.

Top graphs in Figure 2 depicts the mean and standard deviation of the test error and its estimate by each method, as functions of the tuning parameter λ in AIWLS (6). Note that the mean of the test error corresponds to the true risk (see Eqs. 1 and 2). The graphs show that IWCV gives reasonably good unbiased estimates of the risk, while CV is heavily biased.

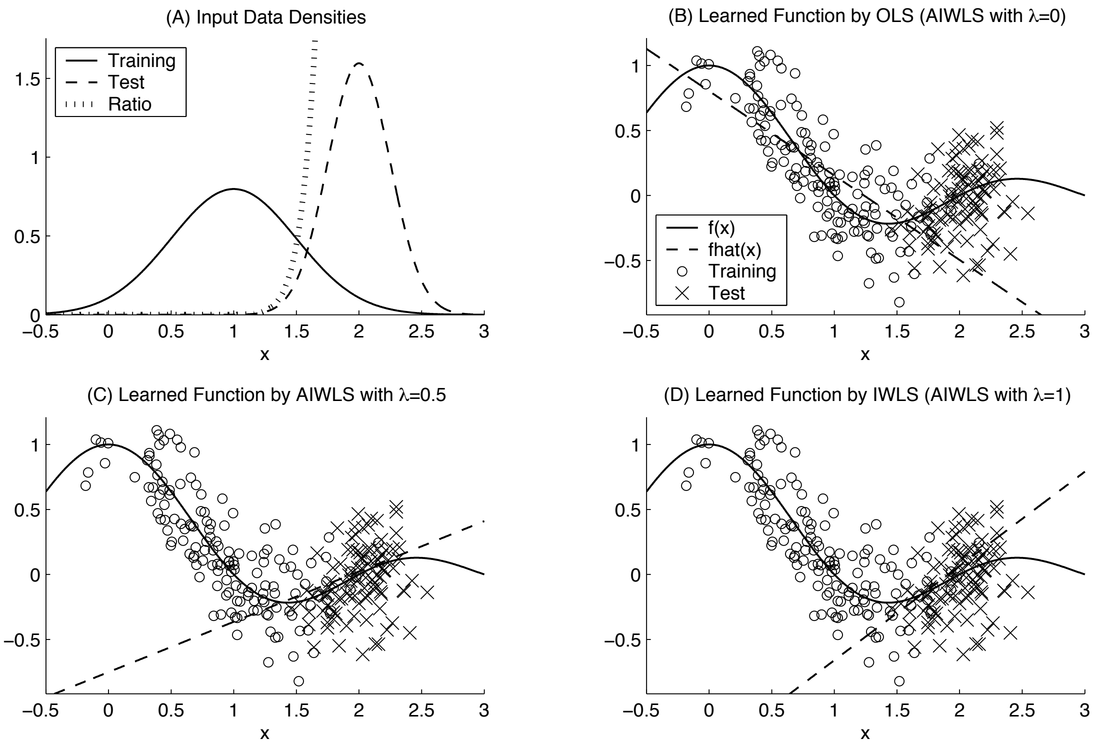


Figure 1: An illustrative regression example of extrapolation by fitting a linear function. (A) The probability density functions of the training and test input points and their ratio. (B)–(D) The learning target function $f(x)$ (the solid line), training samples ('$\circ$'), a learned function $\widehat{f}(x)$ (the dashed line), and test samples ('$\times$'). Note that the test samples are not given in the training phase; they are plotted in the graph for illustration purposes.

Next we investigate the model selection performance: λ is chosen from $\{0, 0.1, 0.2, \ldots, 1\}$ so that the score of each method is minimized. Bottom graphs in Figure 2 depict the histogram of the minimizer of each score. The mean and standard deviation of the test error when λ is chosen by each method are described below the graphs. The numbers show that IWCV gives much smaller test errors than CV (the difference is significant by the *t-test* at the significance level 1%).

We also carried out the same simulation under unknown training and test densities; they are estimated by maximum likelihood fitting of a single Gaussian model or a Gaussian kernel density estimator with variance determined by *Silverman's rule-of-thumb bandwidth selection rule* (Silverman, 1986; Härdle et al., 2004). For estimating the test input density, we draw 100 unlabeled samples following $P_{test}(x)$. The simulation results had very similar trends to the case with known densities (therefore we omit the detail), although the error gets slightly larger. This implies that, for this toy regression problem, estimating the densities from data does not significantly degrade the quality of learning.

The above simulation results illustrate that IWCV performs quite well in covariate-shifted regression tasks.

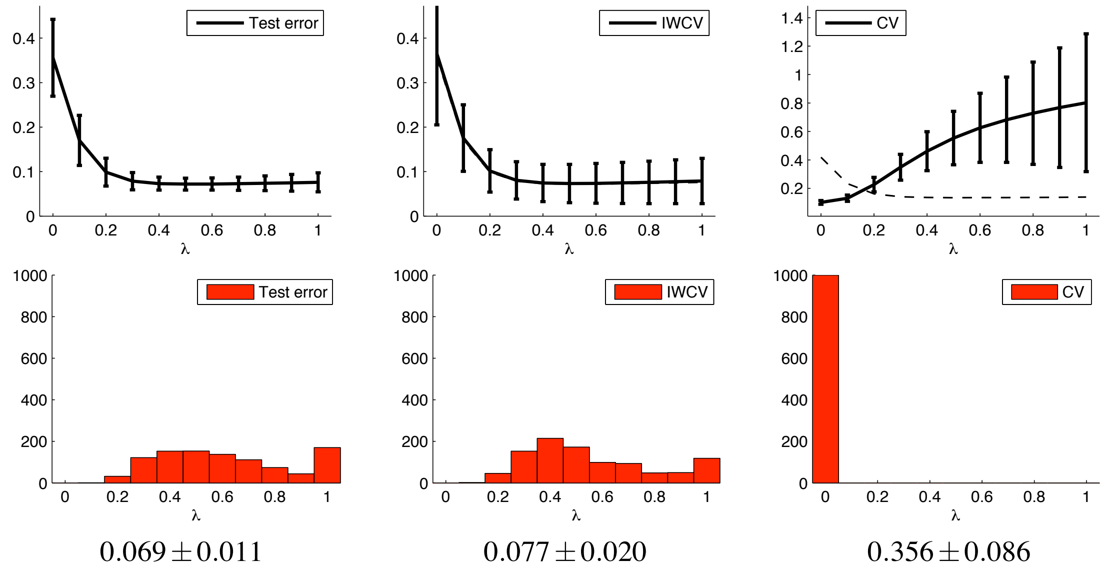


Figure 2: Top graphs: Test error and its estimates as functions of the tuning parameter λ in AIWLS (6). Dashed curves in the middle and right graphs depict the mean test error (i.e., the mean of the left graph) for clear comparison. Bottom graphs: Histograms of the minimizer. The numbers below the graphs are the means and standard deviations of the test error when λ is selected by each method.

4.2 Toy Classification Problem

Through the above regression examples, we found that IWCV works quite well. Here, we apply IWCV to a toy classification problem where the 0/1-loss is used for computing the test error.

Let us consider a binary classification problem on the two-dimensional input space. We define the class posterior probabilities given input x by

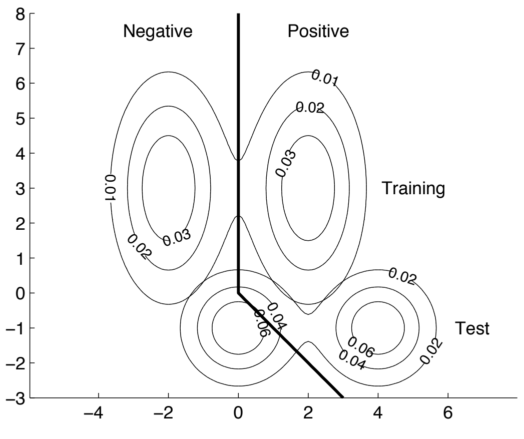
$$p(y=+1|x) = \frac{1+\tanh\left(x^{(1)}+\min(0,x^{(2)})\right)}{2},$$

where $x=(x^{(1)},x^{(2)})^\top$ and $p(y=-1|x)=1-p(y=+1|x)$. The optimal decision boundary, that is, a set of all x such that $p(y=+1|x)=p(y=-1|x)$, is illustrated in Figure 3(b).

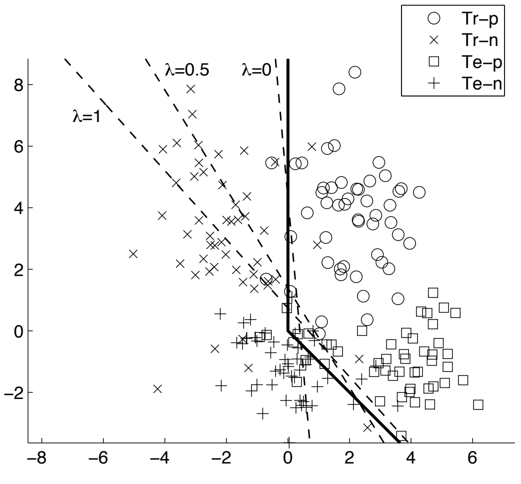
Let the training and test input densities be

$$p_{train}(x) = \frac{1}{2}\phi\left(x;\begin{pmatrix}-2\\3\end{pmatrix},\begin{pmatrix}1&0\\0&2\end{pmatrix}\right)+\frac{1}{2}\phi\left(x;\begin{pmatrix}2\\3\end{pmatrix},\begin{pmatrix}1&0\\0&2\end{pmatrix}\right),$$
$$p_{test}(x) = \frac{1}{2}\phi\left(x;\begin{pmatrix}0\\-1\end{pmatrix},\begin{pmatrix}1&0\\0&1\end{pmatrix}\right)+\frac{1}{2}\phi\left(x;\begin{pmatrix}4\\-1\end{pmatrix},\begin{pmatrix}1&0\\0&1\end{pmatrix}\right),$$

where $\phi(x;\mu,\Sigma)$ is the multivariate normal density with mean μ and covariance matrix Σ. This setting implies that we are considering an extrapolation problem. Contours of the training and test input densities are illustrated in Figure 3(a).



(a) Contours of training and test input densities.



(b) Optimal decision boundary (solid line) and learned boundaries (dashed lines). '$\circ$' and '$\times$' denote the positive and negative training samples, while '$\Box$' and '$+$' denote the positive and negative test samples. Note that the test samples are not given in the training phase; they are plotted in the figure for illustration purposes.

Figure 3: Toy classification problem.

Let $n = 500$ and we create training input points $\{x_i\}_{i=1}^n$ following $P_{train}(x)$ and training output labels $\{y_i\}_{i=1}^n$ following $P(y|x_i)$. Similarly, we create 500 test input points $\{t_i\}_{i=1}^{500}$ following $P_{test}(x)$ and test output labels $\{u_i\}_{i=1}^{500}$ following $P(u|t_i)$.

We use the linear model (5) combined with AIWLS (6) for learning. The classification result $\widehat{u}$ of a test sample t is obtained by the sign of the output of the learned function:

$$\widehat{u} = \text{sgn}\left(\widehat{f}(t;\widehat{\theta}_{AIWLS})\right),$$

where $\text{sgn}(\cdot)$ denotes the sign of a scalar. Note that, if $P_{train}(x) = P_{test}(x)$, this classification method is equivalent to *linear discriminant analysis* (LDA) (Fisher, 1936; Duda et al., 2001), given that the class labels are $y_i \propto \{1/n_+, -1/n_-\}$, where n_+ and n_- are the numbers of positive and negative training samples, respectively. In the following, we rescale the training output values $\{y_i\}_{i=1}^n$ as such, and refer to AIWLS as *adaptive importance weighted LDA* (AIWLDA). Figure 3(b) shows an example of realizations of training and test samples, and decision boundaries obtained by AIWLDA with $\lambda = 0, 0.5, 1$. In this particular realization, $\lambda = 0.5$ or 1 seems to work better than $\lambda = 0$. However, the best value of λ depends on the realization of samples and λ needs to be optimized by a model selection method.

Top graphs in Figure 4 depicts the mean and standard deviation of the test error (i.e., misclassification rate) and its estimate by each method over 1000 runs, as functions of the tuning parameter λ in AIWLDA. The graphs clearly show that IWCV gives much better estimates of the risk than CV.

Next we investigate the model selection performance: λ is chosen from $\{0, 0.1, 0.2, \ldots, 1\}$ so that the score of each method is minimized. Bottom graphs in Figure 4 depict the histograms of

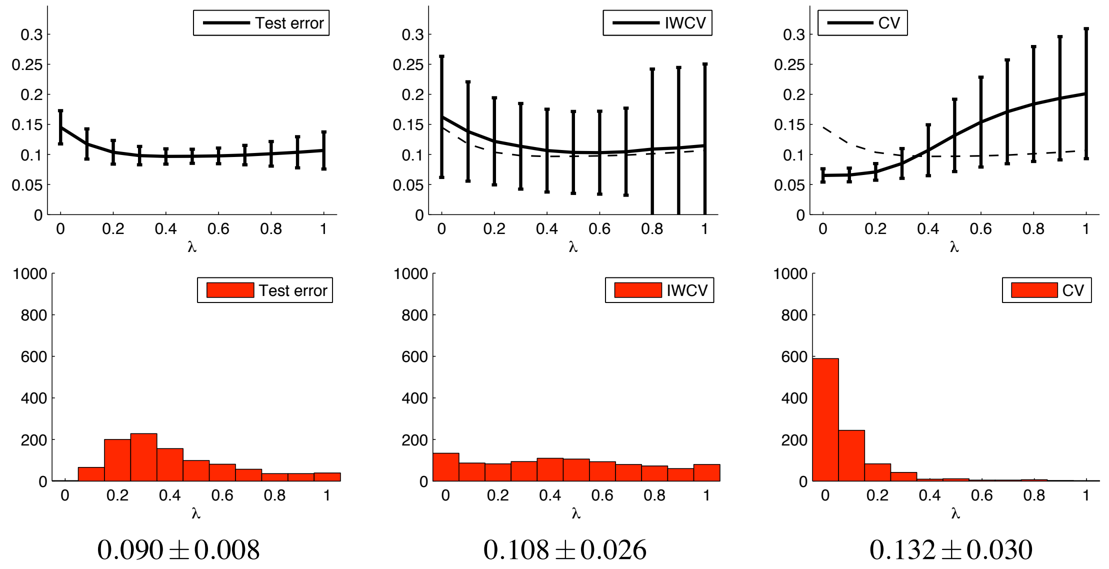


Figure 4: Top graphs: Test error (misclassification rate) and its estimates as functions of the tuning parameter λ in AIWLDA. Dashed curves in the bottom graphs depict the mean test error (i.e., the mean of the left graph) for clear comparison. Bottom graphs: Histograms of the minimizer. The numbers below the graphs are the means and standard deviations of the test error when λ is selected by each method.

the minimizer of each score. The mean and standard deviation of the test error when λ is chosen by each method are described below the graphs. The numbers show that IWCV gives much smaller test errors than CV (the difference is significant by the *t-test* at the significance level 1%).

We also carried out the same simulation except that the training and test densities are unknown; they are estimated by maximum likelihood fitting of a single Gaussian model or a Gaussian kernel density estimator with variance determined by Silverman's rule-of-thumb bandwidth selection rule. For estimating the test input density, we draw 500 unlabeled samples following $P_{test}(x)$. The simulation results were almost identical to the known-density case with a small increase in the error (therefore we omit the detail). This implies that, for this toy classification problem, estimating the densities from data does not significantly degrade the quality of learning.

This simulation result illustrated that IWCV is useful also in covariate-shifted classification tasks.

5. Application to Brain-Computer Interface

The previous section showed that IWCV is promising in both regression and classification tasks with covariate shift. In particular, IWCV is the *only* method which can be successfully applied in covariate-shifted *classification* scenarios. In this section, we apply IWCV to the classification tasks in brain-computer interfaces (BCIs), which attracts a lot of attention these days in biomedical engineering.

A BCI is a system which allows for a direct communication from man to machine (Wolpaw et al., 2002; Dornhege et al., 2007). Cerebral electric activity is recorded via the *electroencephalogram* (EEG): electrodes attached to the scalp measure the electric signals of the brain. These signals are amplified and transmitted to the computer, which translates them into device control commands. The crucial requirement for the successful functioning of BCI is that the electric activity on the scalp surface already reflects motor intentions, that is, the neural correlate of preparation for hand or foot movements. A BCI can detect the motor-related EEG changes and uses this information, for example, to perform a choice between two alternatives: the detection of the preparation to move the left hand leads to the choice of the first control command, whereas the right hand intention would lead to the second alternative. By this means, it is possible to operate devices which are connected to the computer.

For classification of bandpower estimates of appropriately preprocessed EEG signals (Ramoser et al., 2000; Pfurtscheller and da Silva, 1999; Lemm et al., 2005), LDA has shown to work very well (Wolpaw et al., 2002; Dornhege et al., 2004; Babiloni et al., 2000). On the other hand, strong non-stationarity effects have been often observed in brain signals between training and test sessions (Vidaurre et al., 2004; Millán, 2004; Shenoy et al., 2006), which could be regarded as an example of the covariate shift. This indicates that employing importance weighted methods could further improve the BCI recognition accuracy.

Here, we employ AIWLDA to cope with the non-stationarity (see Section 4.2 for detail). We test AIWLDA with totally 14 data sets obtained from 5 different subjects (see Table 1 for specification), where the task is binary classification of EEG signals. The experimental setting is described in more detail in the references (Blankertz et al., 2007, 2006; Sugiyama et al., 2006). Note that training samples and unlabeled/test samples are gathered in different recording sessions, so the non-stationarity in brain signals may change the distributions. On the other hand, the unlabeled samples and test samples are gathered in the same recording session; more precisely, the unlabeled samples are gathered in the first half of the session while the test samples (with labels) are collected in the latter half. Therefore, unlabeled samples may contain some information on the test input distribution, although input distributions of unlabeled and test samples are not necessarily identical since the non-stationarity in brain signals can cause a small change in distributions even within the same session. Thus, this setting realistically renders the classifier update in online BCI systems.

We estimate $p_{train}(x)$ and $p_{test}(x)$ by maximum likelihood fitting of the multi-dimensional Gaussian density with full covariance matrix. $p_{train}(x)$ is estimated using training samples and $p_{test}(x)$ is estimated using the unlabeled samples.

Table 2 describes the misclassification rates of the test samples by LDA (an existing method, which corresponds to AIWLDA with $\lambda = 0$), AIWLDA with λ chosen based on 10-fold IWCV or 10-fold CV, and AIWLDA with optimal λ. The value of λ is selected from $\{0, 0.1, 0.2, \ldots, 1.0\}$; chosen values are also described in the table. Table 2 also contains the *Kullback-Leibler (KL) divergence* (Kullback and Leibler, 1951) from the estimated training input distribution to the estimated test input distribution. Since we want to have an accurate estimate of the KL divergence, we used the test samples for estimating the test input distribution when computing the KL divergence (cf., only unlabeled samples are used when the test input distribution is estimated for AIWLDA and IWCV). The KL values may be roughly interpreted as the *level* of the covariate shift.

First we compare OPT (AIWLDA with optimal λ) with LDA. The table shows that OPT outperforms LDA in 8 out of 14 cases, which motivates us to employ AIWLDA in BCI. Within each subject, we can observe a clear tendency that OPT outperforms LDA when the KL divergence is

Subject	Trial	Dim. of samples	# of training samples	# of unlabeled samples	# of test samples
1	1	3	280	112	112
1	2	3	280	120	120
1	3	3	280	35	35
2	1	3	280	113	112
2	2	3	280	112	112
2	3	3	280	35	35
3	1	3	280	91	91
3	2	3	280	112	112
3	3	3	280	30	30
4	1	6	280	112	112
4	2	6	280	126	126
4	3	6	280	35	35
5	1	2	280	112	112
5	2	2	280	112	112

Table 1: Specification of BCI data.

Subject	Trial	OPT	LDA	IWCV	CV	KL
1	1	* 8.7 (0.5)	9.3 (0)	$^-$ 10.0 (0.9)	10.0 (0.9)	0.76
1	2	* 6.2 (0.3)	8.8 (0)	8.8 (0)	8.8 (0)	1.11
1	3	4.3 (0)	4.3 (0)	4.3 (0)	4.3 (0)	0.69
2	1	40.0 (0)	40.0 (0)	$^\circ$ 40.0 (0)	41.3 (0.7)	0.97
2	2	* 38.7 (0.1)	39.3 (0)	$\overset{+}{\circ}$ 38.7 (0.2)	39.3 (0)	1.05
2	3	25.5 (0)	25.5 (0)	25.5 (0)	25.5 (0)	0.43
3	1	* 34.4 (0.2)	36.9 (0)	$^+$ 34.4 (0.2)	34.4 (0.2)	2.63
3	2	* 18.0 (0.4)	21.3 (0)	$^+$ 19.3 (0.6)	19.3 (0.9)	2.88
3	3	* 15.0 (0.6)	22.5 (0)	$^+$ 17.5 (0.3)	17.5 (0.4)	1.25
4	1	* 20.0 (0.2)	21.3 (0)	21.3 (0)	21.3 (0)	9.23
4	2	2.4 (0)	2.4 (0)	2.4 (0)	2.4 (0)	5.58
4	3	6.4 (0)	6.4 (0)	6.4 (0)	6.4 (0)	1.83
5	1	21.3 (0)	21.3 (0)	21.3 (0)	21.3 (0)	0.79
5	2	* 13.3 (0.5)	15.3 (0)	$\overset{+}{\circ}$ 14.0 (0.1)	15.3 (0)	2.01

Table 2: Misclassification rates for BCI data. All values are in percent. IWCV or CV refers to AIWLDA with λ chosen by 10-fold IWCV or 10-fold CV. OPT refers to AIWLDA with optimal λ. Values of chosen λ are described in the bracket (LDA is denoted as $\lambda = 0$). '$*$' in the table indicates the case where OPT is better than LDA. '$+$' is the case where IWCV outperforms LDA and '$-$' is the opposite case where LDA outperforms IWCV. '$\circ$' denotes the case where IWCV outperforms CV. KL refers to the Kullback-Leibler divergence between (estimated) training and test input distributions.

large, while they are comparable to each other when the KL divergence is small. This well agrees with the theory that AIWLDA can compensate for the effect of the covariate shift, while AIWLDA is reduced to plain LDA in the absence of covariate shift. Next we compare IWCV (applied to AIWLDA) with LDA. IWCV outperforms LDA for 5 cases, while the opposite case occurs only once. The table also shows that within each subject, IWCV tends to outperform LDA when the KL divergence is large. Finally, we compare IWCV with CV (applied to AIWLDA). IWCV outperforms CV for 3 cases, while the opposite case does not occur. IWCV tends to outperform CV when the KL divergence is large within each subject.

6. Discussions, Conclusions, and Future Prospects

In this paper, we discussed the model selection problem under the *covariate shift* paradigm: training input points and test input points are drawn from different distributions (i.e., $P_{train}(x) \neq P_{test}(x)$), but the functional relation remains unchanged (i.e., $P_{train}(y|x) = P_{test}(y|x)$). Under the covariate shift, standard model selection schemes such as cross validation (CV) are heavily biased and do not work as desired. In this paper, we therefore proposed a new variant of CV called importance weighted CV (IWCV), which is proved to be almost unbiased even under the covariate shift.

The model selection problem under the covariate shift has been studied so far. For example, a risk estimator in the context of density estimation called *Akaike's information criterion* (AIC) (Akaike, 1974) was modified to be still asymptotic unbiased (Shimodaira, 2000) and a risk estimator in linear regression called *subspace information criterion* (SIC) (Sugiyama and Ogawa, 2001) was similarly extended to be still unbiased (Sugiyama and Müller, 2005). Although these model selection criteria have rich theoretical properties, the generality of the proposed IWCV goes far beyond them: for the first time arbitrary models, arbitrary parameter learning methods, and arbitrary loss functions can be employed (see Sugiyama et al., 2006, for further discussion and simulation). Thanks to this generality, IWCV enabled us to select appropriate models even in classification tasks under the covariate shift.

We proved that IWCV is almost unbiased even under the covariate shift, which guarantees the quality of IWCV as a risk estimator. However, this does not necessarily imply the good model selection performance since the estimator has some variance. Although our experiments showed that IWCV works well in model selection under the covariate shift, it will be important to also theoretically closer investigate its model selection performance, for example, following the lines of Stone (1977) or Altman and Léger (1997).

In theory, we assumed that the ratio of test and training input densities at training input points is known. On the other hand, they are replaced by appropriate empirical estimates in our simulations. Although the simulation results showed that the proposed method works well even when the densities are unknown, it is valuable to theoretically evaluate the effect of this replacement. Furthermore, developing a more sophisticated method of estimating the density ratio is an important issue to be explored (see, for example, Huang et al., 2007). Feature selection and dimensionality reduction are also important ingredients for better performance. Taking into account the covariate shift in feature selection and dimensionality reduction would be an interesting issue to be pursued, for example, following the lines of He and Niyogi (2004) or Sugiyama (2006b).

While we focused on AIWERM and investigated the model selection performance, developing more sophisticated parameter learning methods would be an important future direction. Graph regularization techniques (Bousquet et al., 2004; Belkin and Niyogi, 2004; Hein, 2006; Chapelle et al., 2006), support vector machines (SVMs) (Vapnik, 1998; Schölkopf and Smola, 2002; Huang et al., 2007), boosting (Schapire, 2003; Meir and Rätsch, 2003) could be useful bases for further development. We note that the proposed IWCV is applicable to any parameter learning methods; even non-parametric learning methods can be employed. Therefore, IWCV may be used for model selection of newly developed learning methods in the future.

We showed experimentally that the IWCV method contributes to improving the accuracy of brain-computer interfaces (BCIs). Future studies along this line will focus on the development of a real time version of the current idea, ultimately striving for fully adaptive learning systems that can appropriately deal with various kinds of non-stationarity. In addition to BCI, there are a number of possible applications, for example, robot control (Shelton, 2001), spam filtering (Bickel and Scheffer, 2007), and bioinformatics (Baldi et al., 1998). Applying IWCV in these application areas would be an interesting direction to be investigated.

Active learning (MacKay, 1992; Cohn et al., 1996; Fukumizu, 2000)—also referred to as *experimental design* in statistics (Kiefer, 1959; Fedorov, 1972; Pukelsheim, 1993)—is the problem of determining the location of training input points $\{x_i\}_{i=1}^n$ so that the risk is minimized. The covariate shift naturally occurs in the active learning scenario since the training input points are generated following a user-defined distribution. For linear regression, IWLS-based active learning methods that focus on minimizing the variance of the estimator have been developed (Wiens, 2000; Sugiyama, 2006a). For general situations including classification with logistic regression models, more elaborated active learning methods which use IWERM have been developed (Kanamori and Shimodaira, 2003; Bach, 2007). In the active learning scenarios, the model has to be fixed, for example, in the above papers, λ in AIWERM (4) is fixed to one. On the other hand, in model selection scenarios, the training input points have to be fixed and corresponding training output values have to be gathered. Thus there exists a dilemma between active learning and model selection (Sugiyama and Ogawa, 2003). An interesting future direction would be to develop a method of performing active learning and model selection at the same time, for example, following the line of Sugiyama and Rubens (2007).

A general situation where the joint training distribution and the joint test distribution are different (i.e., $P_{train}(x,y) \neq P_{test}(x,y)$) is called the *sample selection bias* (see Heckman, 1979). Bayesian generative approaches to coping with such situations have been proposed when unlabeled test input points are available (Storkey and Sugiyama, 2007) or when both test input points and test output values are available (Daumé III and Marcu, 2006). However, due to the Bayesian nature, these approaches implicitly assume that the model used for learning is correctly specified. When this is not true, we may need to reasonably restrict the type of distribution change for meaningful estimations (see, for example, Zadrozny, 2004; Fan et al., 2005; Ben-David et al., 2007; Yamazaki et al., 2007, for theoretical analyses). The covariate shift setting which we discussed in this paper could be regarded as one of such restrictions.

Another interesting restriction on the distribution change is the *class prior probability change* in classification scenarios, where the class prior probabilities are different (i.e., $P_{train}(y) \neq P_{test}(y)$), but the class conditional distribution remains unchanged (i.e., $P_{train}(x|y) = P_{test}(x|y)$). Note that in this case, the functional relation generally changes (i.e., $P_{train}(y|x) \neq P_{test}(y|x)$). SVM (Vapnik, 1998; Schölkopf and Smola, 2002) is a popular classification technique and is shown to converge to the Bayes optimal classifier as the number of training samples tends to infinity (Lin, 2002). However, this nice theoretical property is lost under the class prior probability change. To cope with this problem, SVMs are modified so that the convergence to the Bayes optimal classifier is still guaranteed under the class prior probability change (Lin et al., 2002). Our proposed IWCV idea can be similarly applied in the scenarios of class prior probability change; more specifically, if we replace the importance $p_{test}(x_i)/p_{train}(x_i)$ by $P_{test}(y_i)/P_{train}(y_i)$, IWCV is still almost unbiased even under the class prior probability change. Therefore, IWCV may be used for tuning the model parameters of SVMs even under the class prior probability change. Note that the setting of class prior probability change may be regarded as an extension of *imbalanced classification*, where the ratio of training samples in each class is not even (see, for example, Japkowicz, 2000; Chawla et al., 2003).

Beyond the covariate shift, learning under changing distribution has been gathering significant attention recently (e.g., Bickel, 2006; Candela et al., 2006); note also the large body of work exists in online learning, where the distribution is subject to continuous change (e.g., Robbins and Munro, 1951; Saad, 1998; LeCun et al., 1998; Murata et al., 2002). For further developing learning methods under the changing environment, it is essential to establish and share standard benchmark data sets, for example, the projects supported by PASCAL (Candela et al., 2005) or EPSRC (Kuncheva, 2006), Common benchmark data sets can be used to evaluate the experimental performance of proposed and related methods.

Finally, the importance-weighting idea which was originally used in importance sampling (e.g., Fishman, 1996) could be applied to various statistical procedures, including resampling techniques such as *bootstrap* (Efron, 1979; Efron and Tibshirani, 1993). An interesting future direction is therefore to develop a family of importance-weighted algorithms following the spirit of this paper and to investigate their statistical properties.

Acknowledgments

The authors would like to thank Mitsuo Kawato, Kazuyuki Aihara, Olivier Chapelle, Bernhard Schölkopf, Benjamin Blankertz, Guido Dornhege, Tobias Scheffer, and Masato Okada for their fruitful comments. This work was supported in part by grants of MEXT (Grant-in-Aid for Young Scientists 17700142 and Grant-in-Aid for Scientific Research (B) 18300057) and BMBF (FKZ 01IBE01A/B). A part of this work has been done when MS was staying at University of Edinburgh, which is supported by EU Erasmus Mundus Scholarship. He would like to thank Sethu Vijayakumar for the warm hospitality.

References

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, AC-19(6):716–723, 1974.

N. Altman and C. Léger. On the optimality of prediction-based selection criteria and the convergence rates of estimators. *Journal of the Royal Statistical Society, Series B*, 59(1):205–216, 1997.

F. Babiloni, F. Cincotti, L. Lazzarini, J. d. R. Millán, J. Mourinõ, M. Varsta, J. Heikkonen, L. Bianchi, and M. G. Marciani. Linear classification of low-resolution EEG patterns produced by imagined hand movements. *IEEE Transactions on Rehabilitation Engineering*, 8(2):186–188, June 2000.

F. Bach. Active learning for misspecified generalized linear models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 2007.

P. Baldi, S. Brunak, and G. A. Stolovitzky. *Bioinformatics: The Machine Learning Approach*. MIT Press, Cambridge, 1998.

M. Belkin and P. Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine Learning*, 56(1–3):209–239, 2004.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 2007.

S. Bickel. ECML2006 workshop on discovery challenge, 2006. URL `http://www.ecmlpkdd2006.org/challenge.html`.

S. Bickel and T. Scheffer. Dirichlet-enhanced spam filtering based on biased samples. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 2007.

C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.

B. Blankertz, G. Dornhege, M. Krauledat, and K.-R. Müller. The Berlin brain-computer interface: EEG-based communication without subject training. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14(2):147–152, 2006.

B. Blankertz, G. Dornhege, M. Krauledat, K.-R. Müller, and G. Curio. The non-invasive Berlin brain-computer interface: Fast acquisition of effective performance in untrained subjects. *NeuroImage*, 2007.

K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006.

O. Bousquet, O. Chapelle, and M. Hein. Measure based regularization. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

J. Q. Candela, N. Lawrence, and A. Schwaighofer. Learning when test and training inputs have different distributions challenge, 2005. URL `http://www.pascal-network.org/Challenges/LETTIDD/`.

J. Q. Candela, N. Lawrence, A. Schwaighofer, and M. Sugiyama. NIPS2006 workshop on learning when test and training inputs have different distributions, 2006. URL `http://ida.first.fraunhofer.de/projects/different06/`.

O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, 2006.

N. Chawla, N. Japkowicz, and A. Kolcz. ICML2003 workshop on learning from imbalanced data sets, 2003. URL `http://www.site.uottawa.ca/~nat/Workshop2003/workshop2003.html`.

D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

H. Daumé III and D. Marcu. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126, 2006.

G. Dornhege, B. Blankertz, G. Curio, and K.-R. Müller. Boosting bit rates in non-invasive EEG single-trial classifications by feature combination and multi-class paradigms. *IEEE Transactions on Biomedial Engineering*, 51(6):993–1002, June 2004.

G. Dornhege, J. d. R. Millán, T. Hinterberger, D. McFarland, and K.-R. Müller, editors. *Towards Brain Computer Interfacing*. MIT Press, 2007. in preparation.

R. O. Duda, P. E. Hart, and D. G. Stor. *Pattern Classification*. Wiley, New York, 2001.

B. Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.

B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, New York, 1993.

W. Fan, I. Davidson, B. Zadrozny, and P. S. Yu. An improved categorization of classifier's sensitivity on sample selection bias. In *In Proceedings of the Fifth IEEE International Conference on Data Mining*, 2005.

V. V. Fedorov. *Theory of Optimal Experiments*. Academic Press, New York, 1972.

R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2): 179–188, 1936.

G. S. Fishman. *Monte Carlo: Concepts, Algorithms, and Applications*. Springer-Verlag, Berlin, 1996.

K. Fukumizu. Statistical active learning in multilayer perceptrons. *IEEE Transactions on Neural Networks*, 11(1):17–26, 2000.

W. Härdle, M. Müller, S. Sperlich, and A. Werwatz. *Nonparametric and Semiparametric Models*. Springer Series in Statistics. Springer, Berlin, 2004.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York, 2001.

X. He and P. Niyogi. Locality preserving projections. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

J. J. Heckman. Sample selection bias as a specification error. *Econometrica*, 47(1):153–162, 1979.

M. Hein. Uniform convergence of adaptive graph-based regularization. In *Proceedings of the 19th Annual Conference on Learning Theory*, pages 50–64, 2006.

J. Huang, A. Smola, A. Gretton, K. M. Borgwardt, and B. Schölkopf. Correcting sample selection bias by unlabeled data. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 2007.

N. Japkowicz. AAAI2000 workshop on learning from imbalanced data sets, 2000. URL `http://www.site.uottawa.ca/~nat/Workshop2000/workshop2000.html`.

T. Kanamori and H. Shimodaira. Active learning algorithm using the maximum weighted log-likelihood estimator. *Journal of Statistical Planning and Inference*, 116(1):149–162, 2003.

J. Kiefer. Optimum experimental designs. *Journal of the Royal Statistical Society, Series B*, 21: 272–304, 1959.

S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

L. Kuncheva. Classifiers ensembles for changing environments, 2006. URL `http://gow.epsrc.ac.uk/ViewGrant.aspx?GrantRef=EP/D04040X/1`.

Y. LeCun, L. Bottou, G. B. Orr, and K.-R Müller. Efficient backprop. In G. Orr and K.-R. Müller, editors, *Neural Networks: Tricks of the Trade*, number 1524 in Lecture Notes in Computer Science, pages 299–314. Springer, Berlin, 1998.

S. Lemm, B. Blankertz, G. Curio, and K.-R. Müller. Spatio-spectral filters for improved classification of single trial EEG. *IEEE Transactions on Biomedial Engineering*, 52(9):1541–1548, Sept. 2005.

Y. Lin. Support vector machines and the Bayes rule in classification. *Data Mining and Knowledge Discovery*, 6(3):259–275, 2002.

Y. Lin, Y. Lee, and G. Wahba. Support vector machines for classification in nonstandard situations. *Machine Learning*, 46(1/3):191–202, 2002.

A. Luntz and V. Brailovsky. On estimation of characters obtained in statistical procedure of recognition. *Technicheskaya Kibernetica*, 3, 1969.

D. J. C. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992.

R. Meir and G. Rätsch. An introduction to boosting and leveraging. In S. Mendelson and A. J. Smola, editors, *Advanced Lectures on Machine Learning*, pages 119–184. Springer, Berlin, 2003.

J. d. R. Millán. On the need for on-line learning in brain-computer interfaces. In *Proceedings of the International Joint Conference on Neural Networks*, volume 4, pages 2877–2882, Budapest, Hungary, July 2004.

N. Murata, M. Kawanabe, A. Ziehe, K.-R. Müller, and S. Amari. On-line learning in changing environments with applications in supervised and unsupervised learning. *Neural Networks*, 15 (4-6):743–760, 2002.

G. Pfurtscheller and F. H. Lopes da Silva. Event-related EEG/MEG synchronization and desynchronization: basic principles. *Clinical Neurophysiology*, 110(11):1842–1857, Nov 1999.

F. Pukelsheim. *Optimal Design of Experiments*. Wiley, 1993.

H. Ramoser, J. Müller-Gerking, and G. Pfurtscheller. Optimal spatial filtering of single trial EEG during imagined hand movement. *IEEE Transactions on Rehabilitation Engineering*, 8(4):441–446, 2000.

H. Robbins and S. Munro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

D. Saad, editor. *On-Line Learning in Neural Networks*. Cambridge University Press, Cambridge, 1998.

R. E. Schapire. The boosting approach to machine learning: An overview. In D. D. Denison, M. H. Hansen, C. Holmes, B. Mallick, and B. Yu, editors, *Nonlinear Estimation and Classification*, pages 149–172. Springer, New York, 2003.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

C. R. Shelton. *Importance Sampling for Reinforcement Learning with Multiple Objectives*. PhD thesis, Massachusetts Institute of Technology, 2001.

P. Shenoy, M. Krauledat, B. Blankertz, R. P. N. Rao, and K.-R. Müller. Towards adaptive classification for BCI. *Journal of Neural Engineering*, 3:R13–R23, 2006.

H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244, 2000.

B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, London, 1986.

M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society, Series B*, 36:111–147, 1974.

M. Stone. Asymptotics for and against cross-validation. *Biometrika*, 64(1):29–35, 1977.

A. Storkey and M. Sugiyama. Mixture regression for covariate shift. In B. Schölkopf, J. C. Platt, and T. Hoffmann, editors, *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

M. Sugiyama. Active learning in approximately linear regression based on conditional expectation of generalization error. *Journal of Machine Learning Research*, 7:141–166, Jan. 2006a.

M. Sugiyama. Local Fisher discriminant analysis for supervised dimensionality reduction. In W. Cohen and A. Moore, editors, *Proceedings of 23rd International Conference on Machine Learning*, pages 905–912, Pittsburgh, Pennsylvannia, USA, Jun. 25–29 2006b.

M. Sugiyama, M. Krauledat, and K.-R. Müller. Covariate shift adaptation by importance weighted cross validation. Technical Report TR06-0007, Department of Computer Science, Tokyo Institute of Technology, Sep. 2006. URL `http://www.cs.titech.ac.jp/`.

M. Sugiyama and K.-R. Müller. Input-dependent estimation of generalization error under covariate shift. *Statistics & Decisions*, 23(4):249–279, 2005.

M. Sugiyama and H. Ogawa. Subspace information criterion for model selection. *Neural Computation*, 13(8):1863–1889, 2001.

M. Sugiyama and H. Ogawa. Active learning with model selection—Simultaneous optimization of sample points and models for trigonometric polynomial models. *IEICE Transactions on Information and Systems*, E86-D(12):2753–2763, 2003.

M. Sugiyama and N. Rubens. A batch ensemble approach to active learning with model selection, 2007. (Submitted).

V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

C. Vidaurre, A. Schlögl, R. Cabeza, and G. Pfurtscheller. About adaptive classifiers for brain computer interfaces. *Biomedizinische Technik*, 49(1):85–86, 2004.

G. Wahba. *Spline Model for Observational Data*. Society for Industrial and Applied Mathematics, Philadelphia and Pennsylvania, 1990.

S. Watanabe. Algebraic analysis for nonidentifiable learning machines. *Neural Computation*, 13 (4):899–933, 2001.

D. P. Wiens. Robust weights and designs for biased regression models: Least squares and generalized M-estimation. *Journal of Statistical Planning and Inference*, 83(2):395–412, 2000.

J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan. Brain-computer interfaces for communication and control. *Clinical Neurophysiology*, 113(6):767–791, 2002.

K. Yamazaki, M. Kawanabe, S. Watanabe, M. Sugiyama, and K.-R Müller. Asymptotic Bayesian generalization error when training and test distributions are different, 2007. (Submitted).

B. Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the Twenty-First International Conference on Machine Learning*, New York, NY, 2004. ACM Press.

Journal of Machine Learning Research 8 (2007) 1007-1025 Submitted 10/05; Revised 12/06; Published 5/07

On the Consistency of Multiclass Classification Methods

Ambuj Tewari AMBUJ@CS.BERKELEY.EDU
Division of Computer Science
University of California
Berkeley, CA 94720-1776, USA

Peter L. Bartlett BARTLETT@CS.BERKELEY.EDU
Division of Computer Science and Department of Statistics
University of California
Berkeley, CA 94720-1776, USA

Editor: Peter Auer

Abstract

Binary classification is a well studied special case of the classification problem. Statistical properties of binary classifiers, such as consistency, have been investigated in a variety of settings. Binary classification methods can be generalized in many ways to handle multiple classes. It turns out that one can lose consistency in generalizing a binary classification method to deal with multiple classes. We study a rich family of multiclass methods and provide a necessary and sufficient condition for their consistency. We illustrate our approach by applying it to some multiclass methods proposed in the literature.

Keywords: multiclass classification, consistency, Bayes risk

1. Introduction

We consider the problem of classification in a probabilistic setting: n i.i.d. pairs are generated by a probability distribution on $\mathcal{X} \times \mathcal{Y}$. We think of y_i in a pair (x_i, y_i) as being the *label* or *class* of the example x_i. The $|\mathcal{Y}| = 2$ case is referred to as binary classification. A number of methods for binary classification involve finding a real valued function f which minimizes an empirical average of the form

$$\frac{1}{n} \sum_i \Psi_{y_i}(f(x_i)) \ . \tag{1}$$

In addition, some sort of regularization may be used to avoid overfitting. Typically, the sign of $f(x)$ is used to classify an unseen example x. We interpret $\Psi_y(f(x))$ as being the loss associated with predicting the label of x using $f(x)$ when the true label is y. An important special case of these methods is that of large margin methods which use $\{+1, -1\}$ as the set of labels and $\phi(yf(x))$ as the loss. Here ϕ is some function chosen by taking into account both computational and statistical issues. Bayes consistency of these methods has been analyzed in the literature (see Jiang, 2004; Lugosi and Vayatis, 2004; Zhang, 2004c; Steinwart, 2005; Bartlett et al., 2004). In this paper, we investigate the consistency of multiclass ($|\mathcal{Y}| \geq 2$) methods which try to generalize (1) by replacing f with a vector function $\mathbf{f}$. This category includes the methods of Bredensteiner and Bennett (1999); Lee et al. (2004); Weston and Watkins (1998) and Zhang (2004a). Zhang (2004a,b) has already initiated the study of these methods.

Under suitable conditions, minimizing (1) over a sequence of function classes also approximately minimizes the "Ψ-risk" $R_\Psi(\mathbf{f}) = \mathbb{E}_{\mathcal{X}\mathcal{Y}}[\Psi_y(\mathbf{f}(x))]$. However, our aim in classification is to find a function $\mathbf{f}$ whose probability of misclassification $R(\mathbf{f})$ (often called the "risk" of $\mathbf{f}$) is close to the minimum possible (the so called Bayes risk R^*). Thus, it is natural to investigate the conditions which guarantee that if the Ψ-risk of $\mathbf{f}$ gets close to the optimal then the risk of $\mathbf{f}$ also approaches the Bayes risk. If this happens, we say that the classification method based on Ψ is *Bayes consistent*. Bartlett et al. (2004) defined the notion of "classification calibration" to obtain such conditions for binary classification. The authors also gave a simple characterization of classification calibration for convex loss functions. In Section 1.1 below, we provide a different point of view for looking at classification calibration for binary classification in order to motivate our geometric approach to multiclass classification.

The rest of the paper is organized as follows. Section 2 defines classification calibration in the setting of multiclass classification and provides a justification, in the form of Theorem 2, for studying it: classification calibration is equivalent to Bayes consistency. The main result in Section 3 is Theorem 7 which characterizes classification calibration in terms of geometric properties of some sets associated with the loss function Ψ. In Section 4, we provide certain sufficient conditions for classification calibration. These are provided with the hope that, in practice, some of these conditions will hold and checking the full condition of Theorem 7 can be avoided. Section 5 applies the results obtained in the paper to examine the consistency of a few multiclass methods. Interestingly, many seemingly natural generalizations of binary methods do not lead to consistent multiclass methods. Section 6 provides the conclusion.

1.1 Consistency of Binary Classification Methods

If we have a convex loss function $\phi : \mathbb{R} \mapsto [0,\infty)$ which is differentiable at 0 and $\phi'(0) < 0$, then it is known (Bartlett et al., 2004) that any minimizer f^* of

$$\mathbb{E}_{\mathcal{X}\mathcal{Y}}[\phi(yf(x))] = \mathbb{E}_{\mathcal{X}}[E_{\mathcal{Y}|x}[\phi(yf(x))]] \tag{2}$$

yields a Bayes consistent classifier, that is, $P(Y=+1|X=x) > 1/2 \Rightarrow f^*(x) > 0$ and $P(Y=-1|X=x) < 1/2 \Rightarrow f^*(x) < 0$. In order to motivate the approach of the next section let us work with a few examples. Let us fix an x and denote the two conditional probabilities by p_+ and p_-. We also omit the argument in $f(x)$. We can then write the inner conditional expectation in (2) as

$$p_+\phi(f) + p_-\phi(-f) \; .$$

We wish to find an f which minimizes the expression above. If we define the set $\mathcal{R} \in \mathbb{R}^2$ as

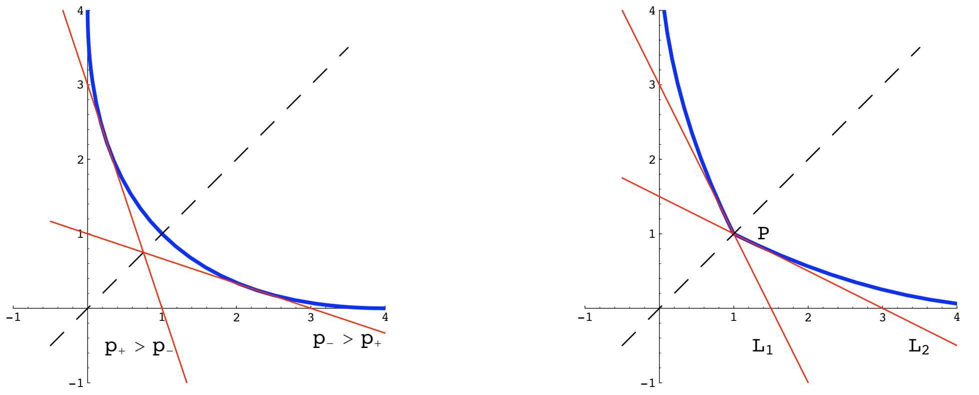
$$\mathcal{R} = \{(\phi(f), \phi(-f)) \; : \; f \in \mathbb{R}\} \; , \tag{3}$$

then the above minimization can be written as

$$\min_{\mathbf{z}\in\mathcal{R}} \langle \mathbf{p}, \mathbf{z} \rangle \tag{4}$$

where $\mathbf{p} = (p_+, p_-)$.

The set $\mathcal{R}$ is shown in Figure 1(a) for the squared hinge loss function $\phi(t) = ((1-t)_+)^2$. Geometrically, the solution to (4) is obtained by taking a line whose equation is $\langle \mathbf{p}, \mathbf{z} \rangle = c$ and then



(a) (b)

Figure 1: (a) Squared Hinge Loss (b) Inconsistent Case (the thick curve is the set $\mathcal{R}$ in both plots)

sliding it (by varying c) until it just touches $\mathcal{R}$. It is intuitively clear from the figure that if $p_+ > p_-$ then the line is inclined more towards the vertical axis and the point of contact is above the angle bisector of the axes. Similarly, if $p_+ < p_-$ then the line is inclined more towards the horizontal axis and the point is below the bisector. This means that $\operatorname{sign}(\phi(-f) - \phi(f))$ is a consistent classification rule which, because ϕ is a decreasing function, is equivalent to $\operatorname{sign}(f - (-f)) = \operatorname{sign}(f)$. In fact, the condition $\phi'(0) < 0$ is not necessary for the existence of a consistent classification rule based on f. For example, if we had the function $\phi(t) = ((1+t)_+)^2$, we would still get the same set $\mathcal{R}$ but will need to change the classification rule to $\operatorname{sign}(-f)$ in order to preserve consistency.

Why do we need differentiability of ϕ at 0? Figure 1(b) shows the set $\mathcal{R}$ for a convex loss function which is not differentiable at 0. In this case, both lines L_1 and L_2 touch $\mathcal{R}$ at P but L_1 has $p_+ > p_-$ while L_2 has $p_+ < p_-$. Thus we cannot create a consistent classifier based on this loss function. The crux of the problem seems to lie in the fact that there are two distinct supporting lines to the set $\mathcal{R}$ at P and that these two lines are inclined towards different axes.

It seems from the figures that as long as $\mathcal{R}$ is symmetric about the angle bisector of the axes, all supporting lines at a given point are inclined towards the same axis except when the point happens to lie on the angle bisector. To check for consistency, we need to examine the set of supporting lines only at that point. In case the set $\mathcal{R}$ is generated as in (3), this boils down to checking the differentiability of ϕ at 0. In the following sections, we will deal with cases when the set $\mathcal{R}$ is generated in a more general way and the situation possibly involves more than two dimensions. The intuition developed for the binary case will be proven correct by Propositions 9 and 10 in Section 4.

2. Classification Calibration and its Relation to Consistency

Suppose we have $K \geq 2$ classes. For $y \in \{1, \ldots, K\}$, let Ψ_y be a continuous function from $\mathbb{R}^K$ to $\mathbb{R}_+ = [0, \infty)$. Let $\mathcal{F}$ be a class of vector functions $\mathbf{f} : \mathcal{X} \mapsto \mathbb{R}^K$. Let $\{\mathcal{F}_n\}$ be a sequence of function classes such that each $\mathcal{F}_n \subseteq \mathcal{F}$. Suppose $\hat{\mathbf{f}}_n$ is a classifier learned from data. For example, we might obtain $\hat{\mathbf{f}}_n$ by minimizing the empirical Ψ-risk $\hat{R}_\Psi$ over the class $\mathcal{F}_n$,

$$\hat{\mathbf{f}}_n = \operatorname*{arg\,min}_{\mathbf{f} \in \mathcal{F}_n} \hat{R}_\Psi(\mathbf{f}) = \operatorname*{arg\,min}_{\mathbf{f} \in \mathcal{F}_n} \frac{1}{n} \sum_{i=1}^{n} \Psi_{y_i}(\mathbf{f}(x_i)) .$$

There might be some constraints on the set of vector functions in the class $\mathcal{F}$. For example, a common constraint is to have the components of $\mathbf{f}$ sum to zero. More generally, let us assume there is some set $\mathcal{C} \in \mathbb{R}^K$ such that

$$\mathcal{F} = \{\mathbf{f} : \forall x,\ \mathbf{f}(x) \in \mathcal{C}\}. \tag{5}$$

Let $\Psi(\mathbf{f}(x))$ denote the vector $(\Psi_1(\mathbf{f}(x)), \ldots, \Psi_K(\mathbf{f}(x)))^T$. We predict the label of a new example x to be $\mathrm{pred}(\Psi(\mathbf{f}(x)))$ for some function $\mathrm{pred} : \mathbb{R}^K \mapsto \{1, \ldots, K\}$. The Ψ-risk of a function $\mathbf{f}$ is

$$R_\Psi(\mathbf{f}) = \mathbb{E}_{\mathcal{X}\mathcal{Y}}[\,\Psi_y(\mathbf{f}(x))\,] ,$$

and we denote the least possible Ψ-risk by

$$R^*_\Psi = \inf_{\mathbf{f} \in \mathcal{F}} R_\Psi(\mathbf{f}) .$$

In a classification task, we are more interested in the risk of a function $\mathbf{f}$,

$$R(\mathbf{f}) = \mathbb{E}_{\mathcal{X}\mathcal{Y}}[\,\mathbf{1}[\mathrm{pred}(\Psi(\mathbf{f}(x))) \neq Y]\,] ,$$

which is the probability that $\mathbf{f}$ leads to an incorrect prediction on a labeled example drawn from the underlying probability distribution. The least possible risk is

$$R^* = \mathbb{E}_{\mathcal{X}}[\,1 - \max_y p_y(x)\,] ,$$

where $p_y(x) = P(Y = y \mid X = x)$. If $\hat{\mathbf{f}}_n$ is the empirical Ψ-risk minimizer then, under suitable conditions, one would expect $R_\Psi(\hat{\mathbf{f}}_n)$ to converge to R^*_Ψ (in probability). It would be nice if that made $R(\hat{\mathbf{f}}_n)$ converge to R^* (in probability). Theorem 2 below states that classification calibration, a notion that we will soon define, is both necessary and sufficient for this to happen.

In order to understand the behavior of approximate Ψ-risk minimizers, let us write $R_\Psi(\mathbf{f})$ as

$$\mathbb{E}_{\mathcal{X}\mathcal{Y}}[\,\Psi_y(\mathbf{f}(x))\,] = \mathbb{E}_{\mathcal{X}}[\,\mathbb{E}_{\mathcal{Y}|x}[\,\Psi_y(\mathbf{f}(x))\,]\,] .$$

The above minimization problem is equivalent to minimizing the inner conditional expectation for each $x \in \mathcal{X}$. Let us fix an arbitrary x for now, so we can write $\mathbf{f}$ instead of $\mathbf{f}(x)$, p_y instead of $p_y(x)$, etc. The minimum might not be achieved and so we consider the infimum[1] $\inf_{\mathbf{f} \in \mathcal{C}} \sum_y p_y \Psi_y(\mathbf{f})$ of the conditional expectation above. Define the subsets $\mathcal{R}$ and $\mathcal{S}$ of $\mathbb{R}^K_+$ as

$$\mathcal{R} = \{(\Psi_1(\mathbf{f}), \ldots, \Psi_K(\mathbf{f})) \ : \ \mathbf{f} \in \mathcal{C}\} ,$$

$$\mathcal{S} = \mathrm{conv}(\mathcal{R}) = \mathrm{conv}\{(\Psi_1(\mathbf{f}), \ldots, \Psi_K(\mathbf{f})) \ : \ \mathbf{f} \in \mathcal{C}\} , \tag{6}$$

where $\mathrm{conv}(\mathcal{R})$ denotes the convex hull of $\mathcal{R}$. We then have

$$\begin{aligned}
\inf_{\mathbf{f} \in \mathcal{C}} \sum_y p_y \Psi_y(\mathbf{f}) &= \inf_{\mathbf{f} \in \mathcal{C}} \langle \mathbf{p}, \Psi(\mathbf{f}) \rangle \\
&= \inf_{\mathbf{z} \in \mathcal{R}} \langle \mathbf{p}, \mathbf{z} \rangle \qquad (7) \\
&= \inf_{\mathbf{z} \in \mathcal{S}} \langle \mathbf{p}, \mathbf{z} \rangle ,
\end{aligned}$$

1. Since p_y and $\Psi_y(\mathbf{f})$ are both non-negative, the objective function is bounded below by 0 and hence the existence of an infimum is guaranteed.

where $\mathbf{p} = (p_1, \dots, p_K)$. The last equality holds because for a fixed $\mathbf{p}$, the function $\mathbf{z} \mapsto \langle \mathbf{p}, \mathbf{z} \rangle$ is a linear function and hence we do not change the infimum by taking the convex hull[2] of $\mathcal{R}$.

Let us define a *symmetric* set to be one with the following property: if a point $\mathbf{z}$ is in the set then so is any point obtained by interchanging any two coordinates of $\mathbf{z}$. We assume that $\mathcal{R}$ is symmetric. This assumption holds whenever the loss function treats all classes equivalently. Note that our assumption about $\mathcal{R}$ implies that $\mathcal{S}$ too is symmetric.

We now define classification calibration of $\mathcal{S}$. The definition intends to capture the property that, for any $\mathbf{p}$, minimizing $\langle \mathbf{p}, \mathbf{z} \rangle$ over $\mathcal{S}$ leads one to $\mathbf{z}$'s which enable us to figure out the index of (one of the) maximum coordinate(s) of $\mathbf{p}$.

Definition 1 *A set $\mathcal{S} \subseteq \mathbb{R}_+^K$ is* **classification calibrated** *if there exists a predictor function* pred : $\mathbb{R}^K \mapsto \{1, \dots, K\}$ *such that*

$$\forall \mathbf{p} \in \Delta_K, \quad \inf_{\mathbf{z} \in \mathcal{S} : \, p_{\mathrm{pred}(\mathbf{z})} < \max_y p_y} \langle \mathbf{p}, \mathbf{z} \rangle > \inf_{\mathbf{z} \in \mathcal{S}} \langle \mathbf{p}, \mathbf{z} \rangle \,, \tag{8}$$

where Δ_K is the probability simplex in $\mathbb{R}^K$.

The following theorem tells us that classification calibration is indeed the right property to study, as it is both necessary and sufficient for convergence of Ψ-risk to the optimal Ψ-risk to imply convergence of risk to the Bayes risk. The convergence of the risk of a classifier to the Bayes risk is often referred to as its consistency.

Theorem 2 *Let Ψ be a (vector-valued) loss function and $\mathcal{C}$ be a subset of $\mathbb{R}^K$. Let $\mathcal{F}$ and $\mathcal{S}$ be as defined in* (5) *and* (6) *respectively. Then $\mathcal{S}$ is classification calibrated iff the following holds. Whenever $\{\mathcal{F}_n\}$ is a sequence of function classes (where $\mathcal{F}_n \subseteq \mathcal{F}$ and $\cup \mathcal{F}_n = \mathcal{F}$) such that $\hat{\mathbf{f}}_n \in \mathcal{F}_n$ and P is the data generating probability distribution,*

$$R_\Psi(\hat{\mathbf{f}}_n) \xrightarrow{P} R_\Psi^*$$

implies

$$R(\hat{\mathbf{f}}_n) \xrightarrow{P} R^* \,.$$

Proof See Appendix A. ■

The definition of classification calibration as stated above is not concrete enough to be of any use in checking the consistency of a multiclass method corresponding to a given loss function. We now study the property of classification calibration with the aim of arriving at a characterization expressed in terms of concretely verifiable properties of the loss function. Before we do that, let us observe that it is easy to reformulate the definition in terms of sequences. The exact statement of the reformulation is provided by the following lemma whose proof, being entirely straightforward, is omitted.

Lemma 3 *$\mathcal{S} \subseteq \mathbb{R}_+^K$ is classification calibrated iff there exists a predictor function* pred : $\mathbb{R}^K \mapsto \{1, \dots, K\}$ *such that the following holds: $\forall \mathbf{p} \in \Delta_K$ and all sequences $\{\mathbf{z}^{(n)}\}$ in $\mathcal{S}$ such that*

$$\langle \mathbf{p}, \mathbf{z}^{(n)} \rangle \to \inf_{\mathbf{z} \in \mathcal{S}} \langle \mathbf{p}, \mathbf{z} \rangle \,, \tag{9}$$

2. If $\mathbf{z}$ is a convex combination of $\mathbf{z}^{(1)}$ and $\mathbf{z}^{(2)}$, then $\langle \mathbf{p}, \mathbf{z} \rangle \geq \min\{\langle \mathbf{p}, \mathbf{z}^{(1)} \rangle, \langle \mathbf{p}, \mathbf{z}^{(2)} \rangle\}$.

we have

$$p_{\mathrm{pred}(\mathbf{z}^{(n)})} = \max_y p_y \tag{10}$$

ultimately.[3]

This makes it easier to see that if $\mathcal{S}$ is classification calibrated then we can find a predictor function such that any sequence achieving the infimum in (7) ultimately predicts the right label (the one having maximum probability). The following lemma shows that symmetry of our set $\mathcal{S}$ allows us to reduce the search space of predictor functions (namely to those functions which map $\mathbf{z}$ to the index of a minimum coordinate).

Lemma 4 *Suppose $\mathcal{S} \subseteq \mathbb{R}^K_+$ is symmetric. If there exists a predictor function* pred *satisfying condition* (8) *in the definition of classification calibration (Definition 1), then any predictor function* pred′ *satisfying*

$$\forall \mathbf{z} \in \mathcal{S},\ z_{\mathrm{pred}'(\mathbf{z})} = \min_y z_y \tag{11}$$

also satisfies (8).

Proof Consider some $\mathbf{p} \in \Delta_K$ and a sequence $\{\mathbf{z}^{(n)}\}$ such that (9) holds. We have $p_{\mathrm{pred}(\mathbf{z}^{(n)})} = \max_y p_y$ ultimately. In order to derive a contradiction, assume that $p_{\mathrm{pred}'(\mathbf{z}^{(n)})} < \max_y p_y$ infinitely often. Since there are finitely many labels, this implies that there is a subsequence $\{\mathbf{z}^{(n_k)}\}$ and labels M and m such that the following hold,

$$\begin{aligned}
\mathrm{pred}(\mathbf{z}^{(n_k)}) = M \in \{y' : p_{y'} = \max_y p_y\}\ , \\
\mathrm{pred}'(\mathbf{z}^{(n_k)}) = m \in \{y' : p_{y'} < \max_y p_y\}\ , \\
\langle \mathbf{p}, \mathbf{z}^{(n_k)} \rangle \to \inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}, \mathbf{z} \rangle\ .
\end{aligned}$$

Because of (11), we also have $z_M^{(n_k)} \geq z_m^{(n_k)}$. Let $\tilde{\mathbf{p}}$ and $\tilde{\mathbf{z}}$ denote the vectors obtained from $\mathbf{p}$ and $\mathbf{z}$ respectively by interchanging the M and m coordinates. Since $\mathcal{S}$ is symmetric, $\mathbf{z} \in \mathcal{S} \Leftrightarrow \tilde{\mathbf{z}} \in \mathcal{S}$. There are two cases depending on whether the inequality in

$$\liminf_k \left(z_M^{(n_k)} - z_m^{(n_k)} \right) \geq 0$$

is strict or not.

If it is strict, denote the value of the lim inf by $\varepsilon > 0$. Then $z_M^{(n_k)} - z_m^{(n_k)} > \varepsilon/2$ ultimately and hence we have

$$\langle \mathbf{p}, \mathbf{z}^{(n_k)} \rangle - \langle \mathbf{p}, \tilde{\mathbf{z}}^{(n_k)} \rangle = (p_M - p_m)(z_M^{(n_k)} - z_m^{(n_k)}) > (p_M - p_m)\varepsilon/2$$

for k large enough. This implies $\liminf_{k\to\infty} \langle \mathbf{p}, \tilde{\mathbf{z}}^{(n_k)} \rangle < \inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}, \mathbf{z} \rangle$, which is a contradiction.

Otherwise, choose a subsequence[4] $\{\mathbf{z}^{(n_k)}\}$ such that $\lim(z_M^{(n_k)} - z_m^{(n_k)}) = 0$. Multiplying this with $(p_M - p_m)$, we have

$$\lim_{k\to\infty} \left(\langle \tilde{\mathbf{p}}, \tilde{\mathbf{z}}^{(n_k)} \rangle - \langle \tilde{\mathbf{p}}, \mathbf{z}^{(n_k)} \rangle \right) = 0\ .$$

3. Throughout the paper, use of “ultimately” in the context of a sequence $\{\mathbf{z}^{(n)}\}$ means “for all sufficiently large values of n”.
4. We do not introduce additional subscripts for simplicity.

We also have

$$\lim_{k\to\infty}\langle \tilde{\mathbf{p}}, \tilde{\mathbf{z}}^{(n_k)}\rangle = \lim_{k\to\infty}\langle \mathbf{p}, \mathbf{z}^{(n_k)}\rangle = \inf_{\mathbf{z}\in\mathcal{S}}\langle \mathbf{p},\mathbf{z}\rangle = \inf_{\mathbf{z}\in\mathcal{S}}\langle \tilde{\mathbf{p}},\tilde{\mathbf{z}}\rangle = \inf_{\mathbf{z}\in\mathcal{S}}\langle \tilde{\mathbf{p}},\mathbf{z}\rangle \ ,$$

where the last equality follows because of symmetry. This means

$$\langle \tilde{\mathbf{p}}, \mathbf{z}^{(n_k)}\rangle \to \inf_{\mathbf{z}\in\mathcal{S}}\langle \tilde{\mathbf{p}},\mathbf{z}\rangle$$

as $k\to\infty$ and therefore

$$\tilde{p}_{\mathrm{pred}(\mathbf{z}^{(n_k)})} = p_M$$

ultimately. This is a contradiction since $\tilde{p}_{\mathrm{pred}(\mathbf{z}^{(n_k)})} = \tilde{p}_M = p_m$. ■

Having identified the class of potentially useful predictor functions, we will henceforth assume that pred is defined as in (11).

3. A Characterization of Classification Calibration

We give a characterization of classification calibration in terms of normals to the convex set $\mathcal{S}$ and its projections onto lower dimensions. For a point $\mathbf{z}\in\partial\mathcal{S}$, we say $\mathbf{p}$ is a normal to $\mathcal{S}$ at $\mathbf{z}$ if[5] $\langle \mathbf{z}'-\mathbf{z},\mathbf{p}\rangle \geq 0$ for all $\mathbf{z}'\in\mathcal{S}$. Define the set of positive normals at $\mathbf{z}$ as

$$\mathcal{N}(\mathbf{z}) = \{\mathbf{p} : \mathbf{p} \text{ is a normal to } \mathcal{S} \text{ at } \mathbf{z}\} \cap \Delta_K \ .$$

Definition 5 *A convex set $\mathcal{S}\subseteq\mathbb{R}^K_+$ is* **admissible** *if $\forall\mathbf{z}\in\partial\mathcal{S}, \forall\mathbf{p}\in\mathcal{N}(\mathbf{z})$, we have*

$$\mathrm{argmin}(\mathbf{z}) \subseteq \mathrm{argmax}(\mathbf{p}) \qquad (12)$$

where $\mathrm{argmin}(\mathbf{z}) = \{y' : z_{y'} = \min_y z_y\}$ *and* $\mathrm{argmax}(\mathbf{p}) = \{y' : p_{y'} = \max_y p_y\}$.

Lemma 6 *If $\mathcal{S}\subseteq\mathbb{R}^K_+$ is admissible then for all $\mathbf{p}\in\Delta_K$ and all* bounded *sequences $\{\mathbf{z}^{(n)}\}$ such that $\langle\mathbf{p},\mathbf{z}^{(n)}\rangle \to \inf_{\mathbf{z}\in\mathcal{S}}\langle\mathbf{p},\mathbf{z}\rangle$, we have $p_{\mathrm{pred}(\mathbf{z}^{(n)})} = \max_y p_y$ ultimately.*

Proof Let $Z(\mathbf{p}) = \{\mathbf{z}\in\partial\mathcal{S} : \mathbf{p}\in\mathcal{N}(\mathbf{z})\}$. Taking the limit of a convergent subsequence of the given bounded sequence gives us a point in $\partial\mathcal{S}$ which achieves the infimum of the inner product with $\mathbf{p}$. Thus, $Z(\mathbf{p})$ is not empty. It is easy to see that $Z(\mathbf{p})$ is closed. For a point $\mathbf{z}$ and a set Z, define

$$\mathrm{dist}(z,Z) = \inf_{\mathbf{z}'\in Z}\|\mathbf{z}-\mathbf{z}'\| \ ,$$

to be the distance of $\mathbf{z}$ from Z. We claim that for all $\varepsilon>0$, $\mathrm{dist}(\mathbf{z}^{(n)},Z(\mathbf{p}))<\varepsilon$ ultimately. For if we assume the contrary, boundedness implies that we can find a convergent subsequence $\{\mathbf{z}^{(n_k)}\}$ such that $\forall k$, $\mathrm{dist}(\mathbf{z}^{(n_k)},Z(\mathbf{p}))\geq\varepsilon$. Let $\mathbf{z}^* = \lim_{k\to\infty}\mathbf{z}^{(n_k)}$. Then $\langle\mathbf{p},\mathbf{z}^*\rangle = \inf_{\mathbf{z}\in\mathcal{S}}\langle\mathbf{p},\mathbf{z}\rangle$ and so $\mathbf{z}^*\in Z(\mathbf{p})$. On the other hand, $\mathrm{dist}(\mathbf{z}^*,Z(\mathbf{p}))\geq\varepsilon$ which gives us a contradiction and our claim is proved.

Now we claim that there exists $\varepsilon'>0$ such that $\mathrm{dist}(\mathbf{z}^{(n)},Z(\mathbf{p}))<\varepsilon'$ implies[6]

$$\mathrm{argmin}(\mathbf{z}^{(n)}) \subseteq \mathrm{argmin}(Z(\mathbf{p})) \ .$$

5. Our sign convention is opposite to the usual one (Rockafellar, 1970, see, for example,) because we are dealing with minimum (instead of maximum) problems.
6. For a set Z, $\mathrm{argmin}(Z)$ denotes $\cup_{\mathbf{z}\in Z}\mathrm{argmin}(\mathbf{z})$.

Assume, for sake of a contradiction, that there is a convergent subsequence $\mathbf{z}^{(n_k)}$ such that

$$\text{dist}(\mathbf{z}^{(n_k)}, Z(\mathbf{p})) \to 0 \text{ as } k \to \infty \ ,$$

and

$$\forall k,\ \text{argmin}(\mathbf{z}^{(n_k)}) \not\subseteq \text{argmin}(Z(\mathbf{p})) \ . \tag{13}$$

Denote the limit by $\mathbf{z}^*$. Since $\text{dist}(\cdot, Z(\mathbf{p}))$ is continuous, $\text{dist}(\mathbf{z}^*, Z(\mathbf{p})) = 0$ which implies that $\mathbf{z}^* \in Z(\mathbf{p})$ as $Z(\mathbf{p})$ is closed. Moreover, for large enough k,

$$\text{argmin}(\mathbf{z}^{(n_k)}) \subseteq \text{argmin}(\mathbf{z}^*) \subseteq \text{argmin}(Z(\mathbf{p})) \ ,$$

where the last inclusion holds because $\mathbf{z}^* \in Z(\mathbf{p})$. This contradicts (13). Hence the claim is proved.

Finally, by admissibility of $\mathcal{S}$, $\text{argmin}(Z(\mathbf{p})) \subseteq \text{argmax}(\mathbf{p})$ and so $\text{argmin}(\mathbf{z}^{(n)}) \subseteq \text{argmax}(\mathbf{p})$ ultimately. ■

The next theorem provides a characterization of classification calibration in terms of normals to $\mathcal{S}$.

Theorem 7 *Let $\mathcal{S} \subseteq \mathbb{R}^K_+$ be a symmetric convex set. Define the projections*

$$\mathcal{S}^{(i)} = \{(z_1, \ldots, z_i)^T : \mathbf{z} \in \mathcal{S}\}$$

for $i \in \{2, \ldots, K\}$. Then $\mathcal{S}$ is classification calibrated iff each $\mathcal{S}^{(i)}$ is admissible.

Proof We prove the easier 'only if' direction first. Suppose some $\mathcal{S}^{(i)}$ is not admissible. Then there exist $\mathbf{z} \in \partial\mathcal{S}^{(i)}$ and $\mathbf{p} \in \mathcal{N}(\mathbf{z})$ and a label y' such that $y' \in \text{argmin}(\mathbf{z})$ and $y' \notin \text{argmax}(\mathbf{p})$. Choose a sequence $\{\mathbf{z}^{(n)}\}$ converging to $\mathbf{z}$. Modify the sequence by replacing, in each $\mathbf{z}^{(n)}$, the coordinates specified by $\text{argmin}(\mathbf{z})$ by their average. The resulting sequence is still in $\mathcal{S}^{(i)}$ (by symmetry and convexity) and has $\text{argmin}(\mathbf{z}^{(n)}) = \text{argmin}(\mathbf{z})$ ultimately. Therefore, if we set $\text{pred}(\mathbf{z}^{(n)}) = y'$, we have $p_{\text{pred}(\mathbf{z}^{(n)})} < \max_y p_y$ ultimately. To get a sequence in $\mathcal{S}$ look at the points whose projections are the $\mathbf{z}^{(n)}$'s and pad $\mathbf{p}$ with $K - i$ zeros.

To prove the other direction, assume each $\mathcal{S}^{(i)}$ is admissible. Consider a sequence $\{\mathbf{z}^{(n)}\}$ with $\langle \mathbf{p}, \mathbf{z}^{(n)} \rangle \to \inf_{\mathbf{z} \in \mathcal{S}} \langle \mathbf{p}, \mathbf{z} \rangle = L$. Without loss of generality, assume that for some $j, 1 \le j \le K$ we have $p_1, \ldots, p_j > 0$ and $p_{j+1}, \ldots, p_K = 0$. We claim that there exists an $M < \infty$ such that $\forall y \le j, z_y^{(n)} \le M$ ultimately. Since $p_j z_j^{(n)} \le L + 1$ ultimately, $M = \max_{1 \le y \le j}\{(L+1)/p_y\}$ works. Consider a set of labels $T \subseteq \{j+1, \ldots, K\}$. Consider the subsequence consisting of those $\mathbf{z}^{(n)}$ for which $z_y^{(n)} \le M$ for $y \in \{1, \ldots, j\} \cup T$ and $z_y^{(n)} > M$ for $y \in \{j+1, \ldots, K\} - T$. The original sequence can be decomposed into finitely many such subsequences corresponding to the $2^{(K-j)}$ choices of the set T. Fix T and convert the corresponding subsequence into a sequence in $\mathcal{S}^{(j+|T|)}$ by dropping the coordinates belonging to the set $\{j+1, \ldots, K\} - T$. Call this sequence $\tilde{\mathbf{z}}^{(n)}$ and let $\tilde{\mathbf{p}}$ be $(p_1, \ldots, p_j, 0, \ldots, 0)^T$. We have a bounded sequence with

$$\langle \tilde{\mathbf{p}}, \tilde{\mathbf{z}}^{(n)} \rangle \to \inf_{\tilde{\mathbf{z}} \in \mathcal{S}^{(j+|T|)}} \langle \tilde{\mathbf{p}}, \tilde{\mathbf{z}} \rangle \ .$$

Thus, by Lemma 6, we have $\tilde{p}_{\text{pred}(\tilde{\mathbf{z}}^{(n)})} = \max_y \tilde{p}_y = \max_y p_y$ ultimately. Since we dropped only those coordinates which were greater than M, $\text{pred}(\tilde{\mathbf{z}}^{(n)})$ picks the same coordinate as $\text{pred}(\mathbf{z}^{(n)})$ where $\mathbf{z}^{(n)}$ is the element from which $\tilde{\mathbf{z}}^{(n)}$ was obtained. Thus we have $p_{\text{pred}(\mathbf{z}^{(n)})} = \max_y p_y$ ultimately and the theorem is proved. ■

4. Sufficient Conditions for Classification Calibration

In this section, we prove some propositions that reduce the work involved in checking classification calibration of sets. We will see that the assumptions made in the propositions are often satisfied. Our first proposition states that in the presence of symmetry, points having a unique minimum coordinate can never destroy admissibility. Before that, we need the following lemma.

Lemma 8 *Let $\mathcal{S} \subseteq \mathbb{R}_+^K$ be a symmetric convex set, $\mathbf{z}$ a point in the boundary of $\mathcal{S}$ and $\mathbf{p} \in \mathcal{N}(\mathbf{z})$. Let $y, y' \in \{1, \ldots, K\}$. Then*

$$(z_{y'} - z_y)(p_y - p_{y'}) \geq 0 .$$

Proof We have $\langle \mathbf{z}' - \mathbf{z}, \mathbf{p} \rangle \geq 0$ for all $\mathbf{z}' \in \mathcal{S}$. Taking limits, this inequality also holds for $\mathbf{z}' \in \partial\mathcal{S}$. Set $\mathbf{z}'$ to be $\mathbf{z}$ with its y, y' coordinates interchanged. By symmetry $\mathbf{z}' \in \partial\mathcal{S}$. Therefore $\langle \mathbf{z}' - \mathbf{z}, \mathbf{p} \rangle \geq 0$ which implies, since $\mathbf{z}, \mathbf{z}'$ agree on all but the y, y' coordinates,

$$(z_{y'} - z_y)p_y + (z_y - z_{y'})p_{y'} \geq 0 .$$

■

Proposition 9 *Let $\mathcal{S} \subseteq \mathbb{R}_+^K$ be a symmetric convex set, $\mathbf{z}$ a point in the boundary of $\mathcal{S}$ and $\mathbf{p} \in \mathcal{N}(\mathbf{z})$. Then* $\operatorname{argmin}(\mathbf{z}) \subseteq \operatorname{argmax}(\mathbf{p})$ *whenever* $|\operatorname{argmin}(\mathbf{z})| = 1$.

Proof For any y, y', Lemma 8 gives us $(z_{y'} - z_y)(p_y - p_{y'}) \geq 0$. Thus $z_y < z_{y'}$ implies $p_y \geq p_{y'}$. Therefore, if $|\operatorname{argmin}(\mathbf{z})| = 1$ and $z_y = \min_{y'} z_{y'}$ then $p_y \geq p_{y'}$ for all y', and so $\operatorname{argmin}(\mathbf{z}) \subseteq \operatorname{argmax}(\mathbf{p})$. ■

If the set $\mathcal{S}$ possesses a unique normal at every point on its boundary then the next proposition guarantees admissibility.

Proposition 10 *Let $\mathcal{S} \subseteq \mathbb{R}_+^K$ be a symmetric convex set, $\mathbf{z}$ a point in the boundary of $\mathcal{S}$ and $\mathcal{N}(\mathbf{z}) = \{\mathbf{p}\}$ is a singleton. Then* $\operatorname{argmin}(\mathbf{z}) \subseteq \operatorname{argmax}(\mathbf{p})$. *Thus, $\mathcal{S}$ is admissible if $|\mathcal{N}(\mathbf{z})| = 1$ for all $\mathbf{z} \in \partial\mathcal{S}$.*

Proof We will assume that there exists a $y, y \in \operatorname{argmin}(\mathbf{z}), y \notin \operatorname{argmax}(\mathbf{p})$ and deduce that there are at least 2 elements in $|\mathcal{N}(\mathbf{z})|$ to get a contradiction. Let $y' \in \operatorname{argmax}(\mathbf{p})$. By Lemma 8 we have $(z_{y'} - z_y)(p_y - p_{y'}) \geq 0$ which implies $z_{y'} \leq z_y$ since $p_y - p_{y'} < 0$. But we already know that $z_y \leq z_{y'}$ and so $z_y = z_{y'}$. Symmetry of $\mathcal{S}$ now implies that $\tilde{\mathbf{p}} \in \mathcal{N}(\mathbf{z})$ where $\tilde{\mathbf{p}}$ is obtained from $\mathbf{p}$ by interchanging the y, y' coordinates. Since $p_y \neq p_{y'}$, $\tilde{\mathbf{p}} \neq \mathbf{p}$ which means $|\mathcal{N}(\mathbf{z})| \geq 2$. ■

Theorem 7 provides a characterization of classification calibration in terms of admissibility of the projections $\mathcal{S}^{(k)}$. As we will see in the examples later, proving that a certain set is not classification calibrated simply involves finding a projection $\mathcal{S}^{(k)}$ and a "bad" point in $\mathcal{S}^{(k)}$ violating the condition of admissibility. On the other hand, the characterization is not so easy to use when we wish to assert, rather than deny, classification calibration. The following lemma shows that under certain assumptions we can ignore the projections $\mathcal{S}^{(k)}$ and only check the admissibility of $\mathcal{S}$. That we cannot always ignore projections will become clear when we consider examples in the next section.

Recall that $\mathcal{R}$ is the set of points $(\Psi_1(\mathbf{f}), \ldots, \Psi_K(\mathbf{f}))$ as $\mathbf{f}$ ranges over $\mathcal{C}$. Define the projections $\mathcal{R}^{(k)}$ in the same way as $\mathcal{S}^{(k)}$. Since the operations of taking projections and convex hulls commute, it is easy to see that $\operatorname{conv}(\mathcal{R}^{(k)}) = \mathcal{S}^{(k)}$.

Theorem 11 *Suppose the set $\mathcal{R}$ is symmetric and the set $\mathcal{S}$ defined in* (6) *is admissible. Then the following holds for any k: if $\mathcal{R}^{(k)}$ is closed then $\mathcal{S}^{(k)}$ is admissible. Furthermore, if $\mathcal{R}^{(k)}$ is closed for all $k \in \{2,\dots,K\}$ then $\mathcal{S}$ is classification calibrated.*

Proof We only need to prove that $\mathcal{R}^{(k)}$ closed and $\mathcal{S}$ admissible implies $\mathcal{S}^{(k)}$ is admissible since the last claim of the theorem follows from this and Theorem 7. Suppose $\mathcal{R}^{(k)}$ is closed and $\mathcal{S}^{(k)}$ is not admissible. Therefore, there exists positive normal $\mathbf{p}$ to $\mathcal{S}^{(k)}$ at a point $\mathbf{z}'$ in the boundary of $\mathcal{S}^{(k)}$ with

$$\operatorname{argmin}(\mathbf{z}') \not\subseteq \operatorname{argmax}(\mathbf{p}) \ . \tag{14}$$

Since $\mathbf{z}'$ is in the boundary of $\mathcal{S}^{(k)}$ and $\operatorname{conv}(\mathcal{R}^{(k)}) = \mathcal{S}^{(k)}$, there exist points $\mathbf{z}^{(1)},\dots,\mathbf{z}^{(l)}$ in the closure of $\mathcal{R}^{(k)}$ such that $\mathbf{p} \in \mathcal{N}(\mathbf{z}^{(i)})$ for all i, and

$$\mathbf{z}' = \sum_{i=1}^{l} \lambda_i \mathbf{z}^{(i)}$$

for some $\lambda_i \geq 0$ and $\sum_i \lambda_i = 1$. It is easy to see that

$$\operatorname{argmin}(\mathbf{z}') \subseteq \bigcup_{i=1}^{l} \operatorname{argmin}(\mathbf{z}^{(i)}) \ .$$

From (14), it now follows that

$$\operatorname{argmin}(\mathbf{z}) \not\subseteq \operatorname{argmax}(\mathbf{p}) \tag{15}$$

holds for some $\mathbf{z} \in \{\mathbf{z}^{(1)},\dots,\mathbf{z}^{(l)}\}$. Now $\mathbf{z}$ is a limit point of $\mathcal{R}^{(k)}$. Since $\mathcal{R}^{(k)}$ is closed, $\mathbf{z} \in \mathcal{R}^{(k)}$. Let $\tilde{\mathbf{z}}$ be the point in $\mathcal{R}$ such that $\mathbf{z} = \operatorname{proj}(\tilde{\mathbf{z}})$ where proj is projection operator mapping $(z_1,\dots,z_K)$ to $(z_1,\dots,z_k)$. Since $\mathcal{R} \subseteq \mathcal{S}$, this gives us $\tilde{\mathbf{z}} \in \mathcal{S}$. Moreover, $\tilde{p}$ is a normal to $\mathcal{S}$ at $\tilde{\mathbf{z}}$ where $\tilde{\mathbf{p}}$ is $\mathbf{p}$ padded with $K-k$ zeros. This is because

$$\langle \tilde{\mathbf{p}}, \mathbf{z}' - \tilde{\mathbf{z}} \rangle = \langle \mathbf{p}, \operatorname{proj}(\mathbf{z}') - \mathbf{z} \rangle \geq 0 \ ,$$

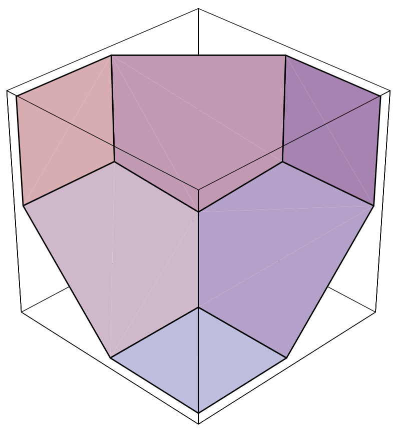
for all $\mathbf{z}' \in \mathcal{S}$. We now claim that $\operatorname{argmin}(\tilde{\mathbf{z}}) \not\subseteq \operatorname{argmax}(\tilde{\mathbf{p}})$, which will give us a contradiction as $\mathcal{S}$ was assumed to be admissible. Since $\tilde{\mathbf{p}}$ is $\mathbf{p}$ padded with zeros, $\operatorname{argmax}(\tilde{\mathbf{p}}) = \operatorname{argmax}(\mathbf{p})$. Also, $\operatorname{argmin}(\tilde{\mathbf{z}}) \subseteq \operatorname{argmin}(\mathbf{z})$ for otherwise applying a suitable permutation to the coordinates of $\tilde{\mathbf{z}}$ will give a point $\hat{\mathbf{z}}$ with $\langle \mathbf{p}, \operatorname{proj}(\hat{\mathbf{z}}) \rangle < \langle \mathbf{p}, \mathbf{z} \rangle$. The claim now follows from (15). ■

We will now provide a sufficient condition for $\mathcal{R}^{(k)}$ to be closed. To state it, we need the following definition.

Definition 12 *Let $\mathcal{C} \subseteq \mathbb{R}^K$ be a set and $\Psi : \mathcal{C} \mapsto \mathbb{R}^K_+$ be a loss function. Further, let $\Psi^{(k)} = (\Psi_1,\dots,\Psi_k)$ be Ψ restricted to the first k coordinates. The mapping $\Psi^{(k)}$ is* **boundedly invertible** *iff for all $M > 0$, there is an $M' > 0$ such that $\|\mathbf{z}\| \leq M$, $\mathbf{z} \in \mathcal{R}^{(k)}$ implies there is $\mathbf{g} \in \mathcal{C}$ with $\|\mathbf{g}\| \leq M'$ and $\Psi^{(k)}(\mathbf{g}) = \mathbf{z}$.*

$\Psi^{(k)}$ boundedly invertible roughly means that it has an inverse carrying bounded points in its range to bounded points in the domain.

Lemma 13 *Suppose that $\mathcal{C}$ is closed and Ψ is continuous. If $\Psi^{(k)}$ is boundedly invertible then $\mathcal{R}^{(k)}$ is closed.*


(a)

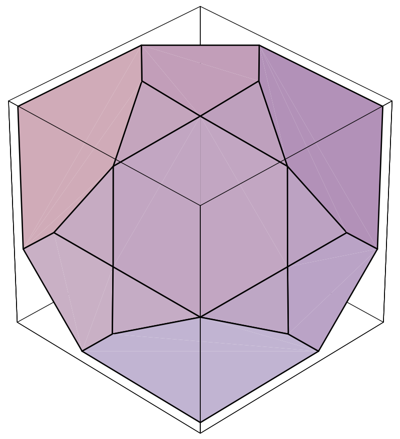

(b)

Figure 2: (a) Crammer and Singer (b) Weston and Watkins

Proof Suppose $\mathbf{z}^{(n)} \in \mathcal{R}^{(k)}$ and $\mathbf{z}^{(n)} \to \mathbf{z}$. Then these is an $M > 0$ such that $\|\mathbf{z}^{(n)}\| \leq M$ for all n. Bounded invertibility implies there is a sequence $\{\mathbf{g}^{(n)}\}$ such that, for all n, $\mathbf{g}^{(n)} \in \mathcal{C}$, $\|\mathbf{g}^{(n)}\| < M'$ and $(\Psi_1(\mathbf{g}^{(n)}), \ldots, \Psi_k(\mathbf{g}^{(n)})) = \mathbf{z}^{(n)}$. Being bounded, $\{\mathbf{g}^{(n)}\}$ has a convergent subsequence $\{\mathbf{g}^{(n_k)}\}$. Since $\mathcal{C}$ is closed, the limit $\mathbf{g}^* \in \mathcal{C}$. Now,

$$\begin{aligned}(\Psi_1(\mathbf{g}^*), \ldots, \Psi_k(\mathbf{g}^*)) &= \lim_{n\to\infty} (\Psi_1(\mathbf{g}^{(n_k)}), \ldots, \Psi_k(\mathbf{g}^{(n_k)})) \text{ (Ψ is continuous)} \\ &= \lim_{n\to\infty} \mathbf{z}^{(n_k)} \\ &= \mathbf{z}\end{aligned}$$

and so $\mathbf{z} \in \mathcal{R}^{(k)}$. ■

5. Examples

We apply the results of the previous section to examine the consistency of several multiclass methods. In all these examples, the functions $\Psi_y(\mathbf{f})$ are obtained from a single real valued function $\psi : \mathbb{R}_+^K \mapsto \mathbb{R}$ as follows

$$\Psi_y(\mathbf{f}) = \psi(f_y, f_1, \ldots, f_{y-1}, f_{y+1}, \ldots, f_K).$$

Moreover, the function ψ is symmetric in its last $K-1$ arguments, that is, interchanging any two of the last $K-1$ arguments does not change the value of the function. This ensures that the set $\mathcal{S}$ is symmetric. We assume that we predict the label of x to be $\arg\min_y \Psi_y(\mathbf{f})$.

5.1 Example 1

The method of Crammer and Singer (2001) corresponds to

$$\Psi_y(\mathbf{f}) = \max_{y' \neq y} \phi(f_y - f_{y'}),\ \mathcal{C} = \mathbb{R}^K$$

with $\phi(t) = (1-t)_+$. For $K = 3$, the boundary of $\mathcal{S}$ is shown in Figure 2(a). At the point $\mathbf{z} = (1,1,1)$, all of these are normals: $(0,1,1)$, $(1,0,1)$, $(1,1,0)$. Thus, there is no y' such that $p_{y'} = \max_y p_y$ for all $\mathbf{p} \in \mathcal{N}(\mathbf{z})$. The method is therefore inconsistent.

Even if we choose an everywhere differentiable convex ϕ with $\phi'(0) < 0$, the three normals mentioned above are still there in $\mathcal{N}(\mathbf{z})$ for $\mathbf{z} = (\phi(0), \phi(0), \phi(0))$. Therefore the method still remains inconsistent.

5.2 Example 2

The method of Weston and Watkins (1998) corresponds to

$$\Psi_y(\mathbf{f}) = \sum_{y' \neq y} \phi(f_y - f_{y'}),\ \mathcal{C} = \mathbb{R}^K \tag{16}$$

with $\phi(t) = (1-t)_+$. For $K = 3$, the boundary of $\mathcal{S}$ is shown in Figure 2(b). The central hexagon has vertices (in clockwise order) $(1,1,4)$, $(0,3,3)$, $(1,4,1)$, $(3,3,0)$, $(4,1,1)$ and $(3,0,3)$. At $\mathbf{z} = (1,1,4)$, we have the following normals: $(1,1,0)$, $(1,1,1)$, $(2,3,1)$, $(3,2,1)$ and there is no coordinate which is maximum in all positive normals. The method is therefore inconsistent.

Now assume that ϕ is a positive convex classification calibrated loss function (i.e., $\phi'(0)$ exists and is negative). Note that if ϕ does not satisfy this assumption then we do not get a consistent method even in the binary classification case. Further assume that ϕ achieves its minimum. The following proposition then guarantees that $\Psi^{(k)}$ is boundedly invertible for $k > 1$ and we can ignore projections. In particular, if we choose ϕ differentiable so that $\mathcal{S}$ possesses a unique normal everywhere on its boundary then, by Proposition 9, $\mathcal{S}$ is admissible and hence classification calibrated. Such is the case if, for example, we choose $\phi(t) = ((1-t)_+)^2$.

Proposition 14 *Suppose $\phi : \mathbb{R} \mapsto [0, \infty)$ is a convex function with $\phi'(0) < 0$. Further, suppose that there is a t such that $\phi(t) = \inf_{t' \in \mathbb{R}} \phi(t')$. Then $\Psi_{(k)}$ (for Ψ given by (16)) is boundedly invertible.*

Proof Since ϕ is a positive convex function with $\phi'(0) < 0$ which achieves its minimum on the real line, only two behaviors are possible for ϕ. Either $\phi(t) \to \infty$ as $t \to \infty$ or there exists $t_0 > 0$ such that $\phi(t)$ is constant for $t \geq t_0$. In the first case, boundedness of $(\Psi_1(\mathbf{f}), \ldots, \Psi_k(\mathbf{f})), k > 1$ implies boundedness of all pairwise differences $f_y - f_{y'}$. Let $m^* = \min_y f_y$ and set $g_y = f_y - m^*$. The value of Ψ_y remains the same as it depends only on differences while all components of $\mathbf{g}$ are now bounded.

In the second case, when we only know that $\phi(t) \to \infty$ as $t \to -\infty$ and that $\phi(t)$ is constant for $t \geq t_0$, boundedness of $(\Psi_1(\mathbf{f}), \ldots, \Psi_k(\mathbf{f}))$ $(k > 1)$, only implies that differences of the form $f_y - f_{y'}$ are bounded from below for $y \in \{1, \ldots, k\}, y' \in \{1, \ldots, K\}$. This implies that $f_y - f_{y'}$ is bounded for $y, y' \leq k$. Let $m^* = \min_{y \leq k} f_y$. Set $h_y = f_y - m^*$ (this doesn't change the value of Ψ as it depends only on differences). Now $h_y \geq 0$ for $y \leq k$ with (at least) one of them exactly zero. Since pairwise differences among these are bounded, the h_y's themselves are bounded for $y \leq k$. This implies $h_{k+1}, \ldots, h_K$ are bounded from above. Now set $g_y = h_y, y \leq k$ and $g_{y'} = \max\{-t_0, h_{y'}\}$ for $y' > k$. To see that we don't change anything by this update, note that when $h_{y'} < -t_0$, both $h_y - h_{y'}$ and $h_y + t_0$ are greater than t_0 (above which ϕ is constant in the case we're considering) for $y \leq k$. Thus, we have managed to make all g_y's bounded without changing the value of Ψ_y for all y. ■

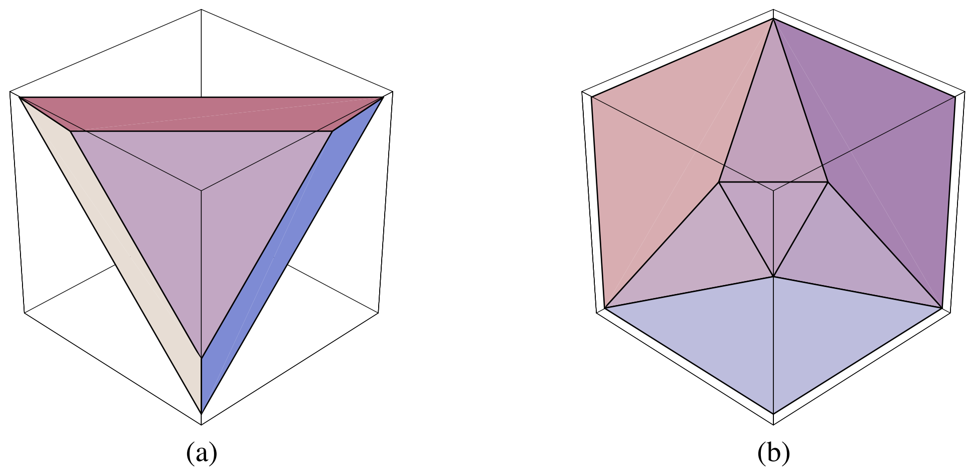

(a) (b)

Figure 3: (a) Lee, Lin and Wahba (b) Loss of consistency in multiclass setting

5.3 Example 3

The method of Lee et al. (2004) corresponds to

$$\Psi_y(\mathbf{f}) = \sum_{y' \neq y} \phi(-f_{y'}),\ \mathcal{C} = \{\mathbf{f} : \sum_y f_y = 0\} \tag{17}$$

with $\phi(t) = (1-t)_+$. Figure 3(a) shows the boundary of $\mathcal{S}$ for $K = 3$. In the general K dimensional case, $\mathcal{S}$ is a polyhedron with K vertices where each vertex has a 0 in one of the positions and K's in the rest. It is obvious then when we minimize $\langle \mathbf{p}, \mathbf{z} \rangle$ over $\mathcal{S}$, we will pick the vertex which has a 0 in the same position where $\mathbf{p}$ has its maximum coordinate. But we can also apply our result here. The set of normals is not a singleton only at the vertices. Thus, by Proposition 10, we only need to check the vertices. Since there is a unique minimum coordinate at the vertices, Proposition 9 implies that the method is consistent.

The question which naturally arises is: for which convex loss functions ϕ does (17) lead to a consistent multiclass classification method? Convex loss functions which are classification calibrated for the two class case, that is, differentiable at 0 with $\phi'(0) < 0$, can lead to inconsistent classifiers of this kind in the multiclass setting. An example is provided by the loss function $\phi(t) = \max\{1-2t, 2-t, 0\}$. Figure 3(b) shows the boundary of $\mathcal{S}$ for $K = 3$. The vertices are $(0,3,3)$, $(9,0,0)$ and their permutations. At $(9,0,0)$, the set of normals includes $(0,1,0)$, $(1,2,2)$ and $(0,0,1)$ and therefore condition (12) is violated.

A nice thing about this example is that if we assume ϕ is a positive classification calibrated binary loss function then $\Psi^{(k)}$ is boundedly invertible for $k > 1$. As in the previous example, this implies that differentiability of ϕ is then sufficient to guarantee consistency. In fact, as Zhang (2004b) shows, a convex function ϕ differentiable on $(-\infty, 0]$ with $\phi'(0) < 0$ will yield a consistent method.

Proposition 15 *Suppose $\phi : \mathbb{R} \mapsto [0,\infty)$ is a convex loss function with $\phi'(0) < 0$. Then $\Psi_{(k)}$ (for Ψ given by* (17)*) is boundedly invertible.*

Proof We have

$$\mathcal{R}^{(k)} = \left\{ \left(\sum_{y' \neq 1} \phi(-f_{y'}), \ldots, \sum_{y' \neq k} \phi(-f_{y'}) \right) : \sum_y f_y = 0 \right\} .$$

We claim that $\phi(t) \to \infty$ as $t \to -\infty$. To see this, note that $\phi'(0) < 0$ so the linear function tangent to ϕ at 0 tends to ∞ as $t \to \infty$. Since ϕ is always above the tangent, the same it true for ϕ. This tells us that each f_y is bounded from above. Since $-f_y = \sum_{y' \neq y} f_{y'}$, $-f_y$ is also bounded from above. This means that each f_y is bounded and $\Psi^{(k)}$ is bounded invertible (just take $\mathbf{g} = \mathbf{f}$ in Definition 12). ■

5.4 Example 4

This is an interesting example because even though we use a differentiable loss function, we still do not have consistency. Let

$$\Psi_y(\mathbf{f}) = \phi(f_y),\ \mathcal{C} = \{\mathbf{f} : \sum_y f_y = 0\}$$

with $\phi(t) = \exp(-\beta t)$ for some $\beta > 0$. One can easily check that

$$\mathcal{R} = \{(z_1, z_2, z_3)^T \in \mathbb{R}^3_+ : z_1 z_2 z_3 = 1\},$$

$$\mathcal{S} = \{(z_1, z_2, z_3)^T \in \mathbb{R}^3_+ : z_1 z_2 z_3 \geq 1\}$$

and so the projection $\mathcal{S}^{(2)}$ is the positive quadrant,

$$\mathcal{S}^{(2)} = \{(z_1, z_2)^T : z_1, z_2 > 0\} .$$

This set is inadmissible and therefore the method is inconsistent.

One can further show that changing ϕ is not of much help. Suppose, ϕ is a positive convex, classification calibrated binary loss function and $K \geq 3$. Then,

$$\mathcal{S}^{(2)} = \{(\phi(f_1), \phi(f_2)) : \sum_y f_y = 0\} .$$

Since $K \geq 3$, (f_1, f_2) is free to assume any value in $\mathbb{R}^2$ and hence $\mathcal{S}^{(2)} = T \times T$ where $T = \{\phi(t) : t \in \mathbb{R}\}$ is the range of ϕ. Let $m = \inf\{\phi(t) : t \in \mathbb{R}\}$. The set T is of the form (m, ∞) or $[m, \infty)$ depending on whether or not ϕ achieves its minimum. Clearly, $\mathcal{S}^{(2)}$ is a translation of the positive quadrant and is not admissible, (m, m) being a point violating the admissibility condition.

5.5 Summary of Examples

Table 1 summarizes how consistency of the four examples treated above depends on the choice of the underlying binary loss function ϕ. The last column gives a condition on ϕ which is sufficient to ensure consistency in case of a convex, positive, classification calibrated ϕ. Note that, for all the examples we considered, mere classification calibration and positivity of ϕ do not suffice to guarantee consistency of the derived multiclass method.

Example	Hinge $(1-t)_+$	Squared Hinge $((1-t)_+)^2$	Exponential $\exp(-t)$	Condition on ϕ
1	$\times$	$\times$	$\times$	Never consistent
2	$\times$	✓	✓[7]	ϕ differentiable & achieves its minimum
3	✓	✓	✓	ϕ differentiable
4	$\times$	$\times$	$\times$	Never consistent

Table 1: Dependence of consistency on the underlying loss function ϕ.

6. Conclusion

We considered multiclass generalizations of classification methods based on convex risk minimization and gave a necessary and sufficient condition for their Bayes consistency. Some examples showed that quite often straightforward generalizations of consistent binary classification methods lead to inconsistent multiclass classifiers. This is especially the case if the original binary method was based on a non-differentiable loss function. Example 4 shows that even differentiable loss functions do not guarantee multiclass consistency. We also showed that in certain cases, one can avoid checking the full set of conditions mentioned in the theorem characterizing classification calibration. The question of consistency then reduces to checking properties of a single convex set. This was illustrated by considering variants of some multiclass methods proposed in the literature and proving their consistency.

Acknowledgments

We gratefully acknowledge the support of NSF under award DMS-0434383 and of ARO under MURI grant DAAD 190210383.

Appendix A.

We will need the following lemma to prove Theorem 2.

Lemma 16 *The function* $\mathbf{p} \mapsto \inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}, \mathbf{z}\rangle$ *is continuous on* Δ_K.

Proof Let $\{\mathbf{p}^{(n)}\}$ be a sequence converging to $\mathbf{p}$. If B is a bounded subset of $\mathbb{R}^K$, then $\langle \mathbf{p}^{(n)}, \mathbf{z}\rangle \to \langle \mathbf{p}, \mathbf{z}\rangle$ uniformly over $\mathbf{z} \in B$ and therefore

$$\inf_{\mathbf{z}\in B} \langle \mathbf{p}^{(n)}, \mathbf{z}\rangle \to \inf_{\mathbf{z}\in B} \langle \mathbf{p}, \mathbf{z}\rangle \ .$$

Let B_r be a ball of radius r in $\mathbb{R}^K$. Then we have

$$\inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}^{(n)}, \mathbf{z}\rangle \le \inf_{\mathcal{S}\cap B_r} \langle \mathbf{p}^{(n)}, \mathbf{z}\rangle \to \inf_{\mathcal{S}\cap B_r} \langle \mathbf{p}, \mathbf{z}\rangle.$$

Therefore

$$\limsup_n \inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}^{(n)}, \mathbf{z}\rangle \le \inf_{\mathbf{z}\in\mathcal{S}\cap B_r} \langle \mathbf{p}, \mathbf{z}\rangle \ .$$

7. This entry does not follow from the results presented in the paper but is included here for completeness.

Letting $r \to \infty$, we get

$$\limsup_n \inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}^{(n)}, \mathbf{z}\rangle \leq \inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}, \mathbf{z}\rangle \ . \tag{18}$$

Without loss of generality, assume that for some $j, 1 \leq j \leq K$ we have $p_1, \dots, p_j > 0$ and $p_{j+1}, \dots, p_K = 0$. For all sufficiently large integers n we can bound the components $p_i^{(n)}$ away from 0 for $i \in \{1, \dots, j\}$. So, for a sufficiently large ball $B_M \subseteq \mathbb{R}^j$ we have

$$\inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}, \mathbf{z}\rangle = \inf_{\mathbf{z}\in\mathcal{S}^{(j)}} \sum_{y=1}^{j} p_y z_y = \inf_{\mathbf{z}\in\mathcal{S}^{(j)}\cap B_M} \sum_{y=1}^{j} p_y z_y \ ,$$

$$\inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}^{(n)}, \mathbf{z}\rangle \geq \inf_{\mathbf{z}\in\mathcal{S}^{(j)}} \sum_{y=1}^{j} p_y^{(n)} z_y = \inf_{\mathbf{z}\in\mathcal{S}^{(j)}\cap B_M} \sum_{y=1}^{j} p_y^{(n)} z_y \ .$$

and thus

$$\liminf_n \inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}^{(n)}, \mathbf{z}\rangle \geq \inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}, \mathbf{z}\rangle \ . \tag{19}$$

Combining (18) and (19), we get

$$\inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}^{(n)}, \mathbf{z}\rangle \to \inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}, \mathbf{z}\rangle \ .$$

■

Theorem 2. *Let Ψ be a (vector-valued) loss function and $\mathcal{C}$ be a subset of $\mathbb{R}^K$. Let $\mathcal{F}$ and $\mathcal{S}$ be as defined in* (5) *and* (6) *respectively. Then $\mathcal{S}$ is classification calibrated iff the following holds. Whenever $\{\mathcal{F}_n\}$ is a sequence of function classes (where $\mathcal{F}_n \subseteq \mathcal{F}$ and $\cup \mathcal{F}_n = \mathcal{F}$) such that $\hat{\mathbf{f}}_n \in \mathcal{F}_n$ and P is the data generating probability distribution,*

$$R_\Psi(\hat{\mathbf{f}}_n) \xrightarrow{P} R_\Psi^*$$

implies

$$R(\hat{\mathbf{f}}_n) \xrightarrow{P} R^* \ .$$

Proof ('only if') Suppose we could prove that $\forall \varepsilon > 0, \exists \delta > 0$ such that $\forall \mathbf{p} \in \Delta_K$,

$$\max_y p_y - p_{\mathrm{pred}(\mathbf{z})} \geq \varepsilon \Rightarrow \langle \mathbf{p}, \mathbf{z}\rangle - \inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}, \mathbf{z}\rangle \geq \delta \ . \tag{20}$$

Using this it immediately follows that $\forall \varepsilon, H(\varepsilon) > 0$ where

$$H(\varepsilon) = \inf_{\mathbf{p}\in\Delta_K, \mathbf{z}\in\mathcal{S}} \{\langle \mathbf{p}, \mathbf{z}\rangle - \inf_{\mathbf{z}\in\mathcal{S}} \langle \mathbf{p}, \mathbf{z}\rangle \,:\, \max_y p_y - p_{\mathrm{pred}(\mathbf{z})} \geq \varepsilon\} \ .$$

A result of Zhang (2004b, Corollary 26) then guarantees there exists a concave function ξ on $[0, \infty)$ such that $\xi(0) = 0$ and $\xi(\delta) \to 0$ as $\delta \to 0^+$ and

$$R(\mathbf{f}) - R^* \leq \xi(R_\Psi(\mathbf{f}) - R_\Psi^*) \ .$$

This inequality combined with $R_\Psi(\hat{\mathbf{f}}_n) \xrightarrow{P} R^*_\Psi$ easily gives $R(\hat{\mathbf{f}}_n) \xrightarrow{P} R^*$. We now prove the implication in (20) by contradiction. Suppose $\mathcal{S}$ is classification calibrated but there exists $\varepsilon > 0$ and a sequence $(\mathbf{z}^{(n)}, \mathbf{p}^{(n)})$ such that

$$p^{(n)}_{\mathrm{pred}(\mathbf{z}^{(n)})} \le \max_y p^{(n)}_y - \varepsilon \tag{21}$$

and

$$\left(\langle \mathbf{p}^{(n)}, \mathbf{z}^{(n)} \rangle - \inf_{\mathbf{z} \in \mathcal{S}} \langle \mathbf{p}^{(n)}, \mathbf{z} \rangle \right) \to 0 \ .$$

Since $\mathbf{p}^{(n)}$ come from a compact set, we can choose a convergent subsequence (which we still denote as $\{\mathbf{p}^{(n)}\}$) with limit $\mathbf{p}$. Using Lemma 16, we get

$$\langle \mathbf{p}^{(n)}, \mathbf{z}^{(n)} \rangle \to \inf_{\mathbf{z} \in \mathcal{S}} \langle \mathbf{p}, \mathbf{z} \rangle \ .$$

As before, we assume that precisely the first j coordinates of $\mathbf{p}$ are non-zero. Then the first j coordinates of $\mathbf{z}^{(n)}$ are bounded for sufficiently large n. Hence

$$\limsup_n \langle \mathbf{p}, \mathbf{z}^{(n)} \rangle = \limsup_n \sum_{y=1}^{j} p^{(n)}_y z^{(n)}_y \le \lim_{n \to \infty} \langle \mathbf{p}^{(n)}, \mathbf{z}^{(n)} \rangle = \inf_{\mathbf{z} \in \mathcal{S}} \langle \mathbf{p}, \mathbf{z} \rangle \ .$$

Now (10) and (21) contradict each other since $\mathbf{p}^{(n)} \to \mathbf{p}$.

('if') If $\mathcal{S}$ is not classification calibrated then by Theorem 7 and Propositions 9 and 10, we have a point in the boundary of some $\mathcal{S}^{(i)}$ where there are at least two normals and which does not have a unique minimum coordinate. Such a point should be there in the projection of $\mathcal{R}$ even without taking the convex hull. Therefore, we must have a sequence $\mathbf{z}^{(n)}$ in $\mathcal{R}$ such that

$$\delta_n = \langle \mathbf{p}, \mathbf{z}^{(n)} \rangle - \inf_{\mathbf{z} \in \mathcal{S}} \langle \mathbf{p}, \mathbf{z} \rangle \to 0 \tag{22}$$

and for all n,

$$p_{\mathrm{pred}(\mathbf{z}^{(n)})} < \max_y p_y \ . \tag{23}$$

Without loss of generality assume that δ_n is a monotonically decreasing sequence. Further, assume that $\delta_n > 0$ for all n. This last assumption might be violated but the following proof then goes through for δ_n replaced by $\max(\delta_n, 1/n)$. Let $\mathbf{g}_n$ be the function that maps every x to one of the pre-images of $\mathbf{z}^{(n)}$ under Ψ. Define $\mathcal{F}_n$ as

$$\mathcal{F}_n = \{\mathbf{g}_n\} \cup (\mathcal{F} \cap \{\mathbf{f} : \forall x, \langle \mathbf{p}, \Psi(\mathbf{f}(x)) \rangle - \inf_{\mathbf{z} \in \mathcal{S}} \langle \mathbf{p}, \mathbf{z} \rangle > 4\delta_n\}$$
$$\cap \{\mathbf{f} : \forall x, \forall j, |\Psi_j(\mathbf{f}(x))| < M_n\})$$

where $M_n \uparrow \infty$ is a sequence which we will fix later. Fix a probability distribution P with arbitrary marginal distribution over x and let the conditional distribution of labels be $\mathbf{p}$ for all x. Our choice of $\mathcal{F}_n$ guarantees that the Ψ-risk of $\mathbf{g}_n$ is less than that of other elements of $\mathcal{F}_n$ by at least $3\delta_n$. Suppose, we make sure that

$$P^n \left(\left| \hat{R}_\Psi(\mathbf{g}_n) - R_\Psi(\mathbf{g}_n) \right| > \delta_n \right) \to 0 \ , \tag{24}$$

$$P^n \left(\sup_{\mathbf{f} \in \mathcal{F}_n - \{\mathbf{g}_n\}} \left| \hat{R}_\Psi(\mathbf{f}) - R_\Psi(\mathbf{f}) \right| > \delta_n \right) \to 0 \ . \tag{25}$$

Then, with probability tending to $1, \hat{\mathbf{f}}_n = \mathbf{g}_n$. By (22), $R_\Psi(\mathbf{g}_n) \to R^*_\Psi$ which implies that $R_\Psi(\hat{\mathbf{f}}_n) \to R^*_\Psi$ in probability. Similarly, (23) implies that $R(\hat{\mathbf{f}}_n) \nrightarrow R^*$ in probability.

We only need to show that we can have (24) and (25) hold. For (24), we apply Chebyshev inequality and use a union bound over the K labels to get

$$P^n \left(\left| \hat{R}_\Psi(\mathbf{g}_n) - R_\Psi(\mathbf{g}_n) \right| > \delta_n \right) \le \frac{K^3 \|\mathbf{z}^{(n)}\|_\infty}{4n\delta_n^2}$$

The right hand side can be made to go to zero by repeating terms in the sequence $\{\mathbf{z}^{(n)}\}$ to slow down the rate of growth of $\|\mathbf{z}^{(n)}\|_\infty$ and the rate of decrease of δ_n. For (24), we use standard covering number bounds (e.g., see Pollard, 1984, Section II.6).

$$P^n \left(\sup_{\mathbf{f} \in \mathcal{F}_n - \{\mathbf{g}_n\}} \left| \hat{R}_\Psi(\mathbf{f}) - R_\Psi(\mathbf{f}) \right| > \delta_n \right) \le 8 \exp \left(\frac{64 M_n^2 \log(2n+1)}{\delta_n^2} - \frac{n\delta_n^2}{128 M_n^2} \right)$$

Thus M_n/δ_n needs to grow slowly enough such that

$$\frac{n\delta_n^4}{M_n^4 \log(2n+1)} \to \infty \, .$$

■

References

Peter L. Bartlett, Michael I. Jordan, and Jon D. McAuliffe. Large margin classifiers: Convex loss, low noise and convergence rates. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2004.

Erin J. Bredensteiner and Kristin P. Bennett. Multicategory classification by support vector machines. *Computational Optimization and Applications*, 12:35–46, 1999.

Koby Crammer and Yoram Singer. On the algorithmic implementation of kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

Wenxin Jiang. Process consistency for AdaBoost. *Annals of Statistics*, 32(1):13–29, 2004.

Yoonkyung Lee, Yi Lin, and Grace Wahba. Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99(465):67–81, 2004.

Gábor Lugosi and Nicolas Vayatis. On the Bayes-risk consistency of regularized boosting methods. *Annals of Statistics*, 32(1):30–55, 2004.

David Pollard. *Convergence of Stochastic Processes*. Springer-Verlag, New York, 1984.

R. Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, NJ, 1970.

Ingo Steinwart. Consistency of support vector machines and other regularized kernel classifiers. *IEEE Transactions on Information Theory*, 51(1):128–142, 2005.

Jason Weston and Chris Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Department of Computer Science, Royal Holloway College, University of London, 1998.

Tong Zhang. An infinity-sample theory for multi-category large margin classification. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2004a.

Tong Zhang. Statistical analysis of some multi-category large margin classification methods. *Journal of Machine Learning Research*, 5:1225–1251, 2004b.

Tong Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals of Statistics*, 32(1):56–85, 2004c.

Journal of Machine Learning Research 8 (2007) 1027-1061 Submitted 3/06; Revised 12/06; Published 5/07

Dimensionality Reduction of Multimodal Labeled Data by Local Fisher Discriminant Analysis*

Masashi Sugiyama SUGI@CS.TITECH.AC.JP
Department of Computer Science
Tokyo Institute of Technology
2-12-1, O-okayama, Meguro-ku, Tokyo, 152-8552, Japan

Editor: Sam Roweis

Abstract

Reducing the dimensionality of data without losing intrinsic information is an important preprocessing step in high-dimensional data analysis. Fisher discriminant analysis (FDA) is a traditional technique for supervised dimensionality reduction, but it tends to give undesired results if samples in a class are *multimodal*. An unsupervised dimensionality reduction method called locality-preserving projection (LPP) can work well with multimodal data due to its locality preserving property. However, since LPP does not take the label information into account, it is not necessarily useful in supervised learning scenarios. In this paper, we propose a new linear supervised dimensionality reduction method called *local Fisher discriminant analysis* (LFDA), which effectively combines the ideas of FDA and LPP. LFDA has an analytic form of the embedding transformation and the solution can be easily computed just by solving a generalized eigenvalue problem. We demonstrate the practical usefulness and high scalability of the LFDA method in data visualization and classification tasks through extensive simulation studies. We also show that LFDA can be extended to non-linear dimensionality reduction scenarios by applying the kernel trick.

Keywords: dimensionality reduction, supervised learning, Fisher discriminant analysis, locality preserving projection, affinity matrix

1. Introduction

The goal of dimensionality reduction is to embed high-dimensional data samples in a low-dimensional space so that most of 'intrinsic information' contained in the data is preserved (e.g., Roweis and Saul, 2000; Tenenbaum et al., 2000; Hinton and Salakhutdinov, 2006). Once dimensionality reduction is carried out appropriately, the compact representation of the data can be used for various succeeding tasks such as visualization, classification, etc. In this paper, we consider the *supervised* dimensionality reduction problem, that is, samples are accompanied with class labels.

Fisher discriminant analysis (FDA) (Fisher, 1936; Fukunaga, 1990) is a popular method for linear supervised dimensionality reduction.[1] FDA seeks for an embedding transformation such

$*$. An efficient MATLAB implementation of local Fisher discriminant analysis is available from the author's website: '`http://sugiyama-www.cs.titech.ac.jp/~sugi/software/LFDA/`'.

1. FDA may refer to the classification method which first projects data samples onto a one-dimensional subspace and then classifies the samples by thresholding (Fisher, 1936; Duda et al., 2001). The one-dimensional embedding space used here is obtained as the maximizer of the so-called *Fisher criterion*. This Fisher criterion can be used for dimensionality reduction onto a space with dimension more than one in multi-class problems (Fukunaga, 1990). With some abuse, we refer to the dimensionality reduction method based on the Fisher criterion as FDA (see Section 2.2 for detail).

that the between-class scatter is maximized and the within-class scatter is minimized. FDA is a traditional but useful method for dimensionality reduction. However, it tends to give undesired results if samples in a class form several separate clusters (i.e., *multimodal*) (see, e.g., Fukunaga, 1990).

Within-class multimodality can be observed in many practical applications. For example, in disease diagnosis, the distribution of medial checkup samples of sick patients could be multimodal since there may be several different causes even for a single disease. In a traditional task of hand-written digit recognition, within-class multimodality appears if digits are classified into, for example, even and odd numbers. More generally, solving multi-class classification problems by a set of two-class 'one-versus-rest' problems naturally induces within-class multimodality. For this reason, there is a universal need for reducing the dimensionality of multimodal data.

In order to reduce the dimensionality of multimodal data appropriately, it is important to preserve the local structure of the data. *Locality-preserving projection* (LPP) (He and Niyogi, 2004) meets this requirement; LPP seeks for an embedding transformation such that nearby data pairs in the original space close in the embedding space. Thus LPP can reduce the dimensionality of multimodal data without losing the local structure. However, LPP is an unsupervised dimensionality reduction method and does not take the label information into account. Therefore, it does not necessarily work appropriately in supervised dimensionality reduction scenarios.

In this paper, we propose a new dimensionality reduction method called *local Fisher discriminant analysis* (LFDA). LFDA effectively combines the ideas of FDA and LPP, that is, LFDA maximizes between-class separability and preserves *within-class local structure* at the same time. Thus LFDA is useful for dimensionality reduction of multimodal labeled data.

The original FDA provides a meaningful result only when the dimensionality of the embedding space is smaller than the number of classes because of the rank deficiency of the between-class scatter matrix (Fukunaga, 1990). This is an essential limitation of FDA in dimensionality reduction. On the other hand, the proposed LFDA does not generally suffer from this problem and can be employed for dimensionality reduction into an *arbitrary* dimensional space. Furthermore, LFDA inherits an excellent property from FDA—it has an *analytic* form of the embedding matrix and the solution can be easily computed just by solving a generalized eigenvalue problem. This is an advantage over recently proposed supervised dimensionality reduction methods (e.g., Goldberger et al., 2005; Globerson and Roweis, 2006). Furthermore, LFDA can be naturally extended to non-linear dimensionality reduction scenarios by applying the *kernel trick* (Schölkopf and Smola, 2002).

The rest of this paper is organized as follows. In Section 2, we formulate the linear dimensionality reduction problem, briefly review FDA and LPP, and illustrate how they typically behave. In Section 3, we define LFDA and show its fundamental properties. In Section 4, we discuss the relation between LFDA and other methods. In Section 5, we numerically evaluate the performance of LFDA and existing methods in visualization and classification tasks using benchmark data sets. Finally, we give concluding remarks and future prospects in Section 6.

2. Linear Dimensionality Reduction

In this section, we formulate the problem of linear dimensionality reduction and review existing methods.

2.1 Formulation

Let $x_i \in \mathbb{R}^d$ $(i = 1, 2, \ldots, n)$ be d-dimensional samples and $y_i \in \{1, 2, \ldots, c\}$ be associated class labels, where n is the number of samples and c is the number of classes. Let n_ℓ be the number of samples in class ℓ:

$$\sum_{\ell=1}^{c} n_\ell = n.$$

Let X be the matrix of all samples:

$$X \equiv (x_1 | x_2 | \cdots | x_n).$$

Let $z_i \in \mathbb{R}^r$ $(1 \leq r \leq d)$ be low-dimensional representations of x_i, where r is the reduced dimension (i.e., the dimension of the embedding space). Effectively we consider d to be large and r to be small, but not limited to such cases.

For the moment, we focus on linear dimensionality reduction, that is, using a $d \times r$ transformation matrix T, the embedded samples z_i are given by

$$z_i = T^\top x_i,$$

where $^\top$ denotes the transpose of a matrix or vector. In Section 3.4, we extend our discussion to the non-linear dimensionality reduction scenarios where the mapping from x_i to z_i is non-linear.

2.2 Fisher Discriminant Analysis for Dimensionality Reduction

One of the most popular dimensionality reduction techniques is *Fisher discriminant analysis* (FDA) (Fisher, 1936; Fukunaga, 1990; Duda et al., 2001). Here we briefly describe the definition of FDA.

Let $S^{(w)}$ and $S^{(b)}$ be the *within-class scatter matrix* and the *between-class scatter matrix*:

$$S^{(w)} \equiv \sum_{\ell=1}^{c} \sum_{i:y_i=\ell} (x_i - \mu_\ell)(x_i - \mu_\ell)^\top, \tag{1}$$

$$S^{(b)} \equiv \sum_{\ell=1}^{c} n_\ell (\mu_\ell - \mu)(\mu_\ell - \mu)^\top, \tag{2}$$

where $\sum_{i:y_i=\ell}$ denotes the summation over i such that $y_i = \ell$, μ_ℓ is the mean of the samples in class ℓ, and μ is the mean of all samples:

$$\mu_\ell \equiv \frac{1}{n_\ell} \sum_{i:y_i=\ell} x_i,$$

$$\mu \equiv \frac{1}{n} \sum_{i=1}^{n} x_i = \frac{1}{n} \sum_{\ell=1}^{c} n_\ell \mu_\ell.$$

We assume that $S^{(w)}$ has full rank. The FDA transformation matrix T_{FDA} is defined as follows:[2]

$$T_{FDA} \equiv \operatorname*{argmax}_{T \in \mathbb{R}^{d\times r}} \left[\operatorname{tr}\left((T^\top S^{(w)} T)^{-1} T^\top S^{(b)} T \right) \right]. \tag{3}$$

2. The following definition is also used in the literature (e.g., Fukunaga, 1990) and yields the same solution.

$$T_{FDA} = \operatorname*{argmax}_{T \in \mathbb{R}^{d\times r}} \left[\frac{\det\left(T^\top S^{(b)} T\right)}{\det\left(T^\top S^{(w)} T\right)} \right],$$

where $\det(\cdot)$ denotes the determinant of a matrix.

That is, FDA seeks a transformation matrix T such that the between-class scatter is 'maximized' while the within-class scatter is 'minimized'. In the above formulation, we implicitly assumed that $T^\top S^{(w)} T$ is invertible. This implies that the above optimization is subject to

$$\mathrm{rank}(T) = r.$$

Let $\{\varphi_k\}_{k=1}^d$ be the generalized eigenvectors associated with the generalized eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$ of the following generalized eigenvalue problem:

$$S^{(b)}\varphi = \lambda S^{(w)}\varphi.$$

Then a solution T_{FDA} of the above maximization problem is analytically given by

$$T_{FDA} = (\varphi_1|\varphi_2|\cdots|\varphi_r).$$

Note that the solution is not unique and the following simple constraint is sometimes imposed additionally (Fukunaga, 1990).

$$T_{FDA}^\top S^{(w)} T_{FDA} = I_r,$$

where I_r is the identity matrix on $\mathbb{R}^r$. This constraint makes the within-class scatter in the embedding space *sphered*.

The between-class scatter matrix $S^{(b)}$ has at most rank $c-1$ (Fukunaga, 1990). This implies that the multiplicity of $\lambda = 0$ is at least $d-c+1$. Therefore, FDA can find at most $c-1$ meaningful features; the remaining features found by FDA are arbitrary. This is an essential limitation of FDA for dimensionality reduction and is very restrictive in practice.

2.3 Locality-Preserving Projection

Another dimensionality reduction technique that is relevant to the current setting is *locality-preserving projection* (LPP) (He and Niyogi, 2004). Here we review LPP.

Let A be an *affinity matrix*, that is, the n-dimensional matrix with the (i,j)-th element $A_{i,j}$ being the affinity between x_i and x_j. We assume that $A_{i,j} \in [0,1]$; $A_{i,j}$ is large if x_i and x_j are 'close' and $A_{i,j}$ is small if x_i and x_j are 'far apart'. There are several different manners of defining A. We briefly describe typical definitions in Appendix D. The LPP transformation matrix T_{LPP} is defined as follows:[3]

$$T_{LPP} \equiv \mathop{\mathrm{argmin}}_{T\in\mathbb{R}^{d\times r}} \left(\frac{1}{2}\sum_{i,j=1}^n A_{i,j} \|T^\top x_i - T^\top x_j\|^2 \right)$$
$$\text{subject to } T^\top X D X^\top T = I_r, \tag{4}$$

where D is the n-dimensional diagonal matrix with i-th diagonal element being

$$D_{i,i} \equiv \sum_{j=1}^n A_{i,j}.$$

3. The matrix D in the constraint (4) is motivated by a geometric argument (Belkin and Niyogi, 2003). However, it is sometimes dropped for the sake of simplicity (Ham et al., 2004).

Eq. (4) implies that LPP looks for a transformation matrix T such that *nearby* data pairs in the original space $\mathbb{R}^d$ are kept close in the embedding space. The constraint (4) is imposed for avoiding degeneracy.

Let $\{\psi_k\}_{k=1}^{d}$ be the generalized eigenvectors associated with the generalized eigenvalues $\gamma_1 \geq \gamma_2 \geq \cdots \geq \gamma_d$ of the following generalized eigenvalue problem:

$$XLX^\top \psi = \gamma XDX^\top \psi,$$

where

$$L \equiv D - A.$$

L is called the *graph-Laplacian matrix* in the spectral graph theory (Chung, 1997), where A is seen as the *adjacency matrix* of a graph. He and Niyogi (2004) showed that a solution of Eq. (4) is given by

$$T_{LPP} = (\psi_d | \psi_{d-1} | \cdots | \psi_{d-r+1}).$$

2.4 Typical Behavior of FDA and LPP

Dimensionality reduction results obtained by FDA and LPP are illustrated in Figure 1 (LFDA will be defined and explained in Section 3)—two-dimensional two-class data samples are embedded into a one-dimensional space. In LPP, the affinity matrix A is determined by the *local scaling method* (Zelnik-Manor and Perona, 2005, see also Appendix D.4).

For the simplest data set depicted in Figure 1(a), both FDA and LPP nicely separate the samples in different classes ('$\circ$' and '$\times$') from each other. For the data set depicted in Figure 1(b), FDA still works well, but LPP mixes samples in different classes into a single cluster. This is caused by the unsupervised nature of LPP. On the other hand, for the data set depicted in Figure 1(c), LPP works well but FDA collapses the samples in different classes into a single cluster. The reason for the failure of FDA is that the 'levels' of the between-class scatter and the within-class scatter are not evaluated in an intuitively natural way because of the two separate clusters in '$\circ$'-class (see also Fukunaga, 1990).

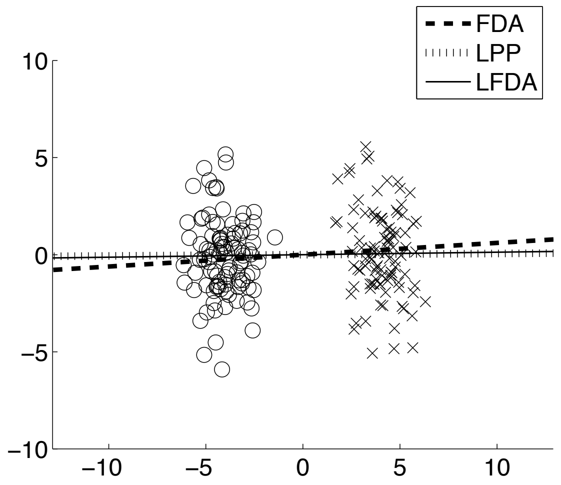
3. Local Fisher Discriminant Analysis

As illustrated in Figure 1, FDA can perform poorly if samples in a class form several separate clusters (i.e., *multimodal*). In other words, the undesired behavior of FDA is caused by the *globality* when evaluating the within-class scatter and the between-class scatter (e.g., Figure 1(c)). On the other hand, because of the unsupervised nature of LPP, it can overlap samples in different classes if they are close in the original high-dimensional space $\mathbb{R}^d$ (e.g., Figure 1(b)).

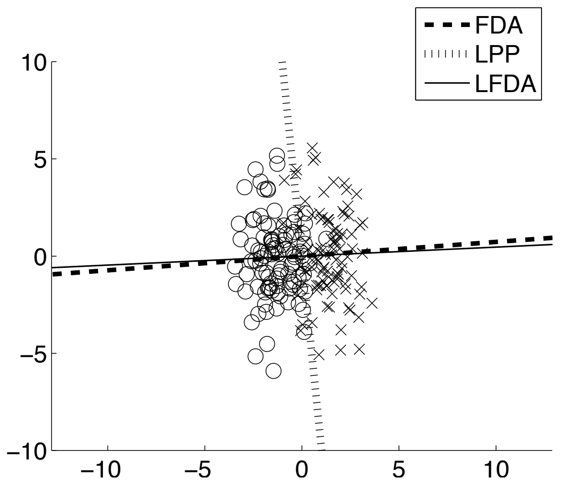
To overcome these problems, we propose combining the ideas of FDA and LPP; more specifically, we evaluate the levels of the between-class scatter and the within-class scatter in a *local* manner. This allows us to attain between-class separation and within-class local structure preservation at the same time. We call our new method *local Fisher discriminant analysis* (LFDA).

3.1 Reformulating FDA

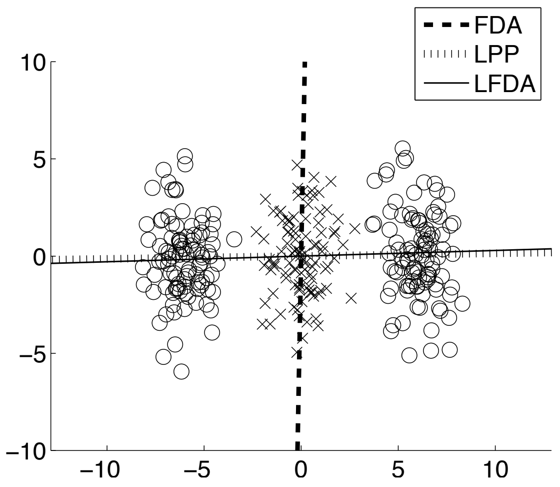
In order to introduce LFDA, let us first reformulate FDA in a *pairwise* manner.



(a) Toy data set 1



(b) Toy data set 2



(c) Toy data set 3

Figure 1: Examples of dimensionality reduction by FDA, LPP and LFDA. Two-dimensional two-class samples are embedded into a one-dimensional space. The line in the figure denotes the one-dimensional embedding space (which the data samples are projected on) obtained by each method.

Lemma 1 *$S^{(w)}$ and $S^{(b)}$ defined by Eqs.* (1) *and* (2) *can be expressed as*

$$S^{(w)} = \frac{1}{2}\sum_{i,j=1}^{n} W^{(w)}_{i,j}(x_i - x_j)(x_i - x_j)^\top, \tag{5}$$

$$S^{(b)} = \frac{1}{2}\sum_{i,j=1}^{n} W^{(b)}_{i,j}(x_i - x_j)(x_i - x_j)^\top, \tag{6}$$

where

$$W^{(w)}_{i,j} \equiv \begin{cases} 1/n_\ell & \text{if } y_i = y_j = \ell, \\ 0 & \text{if } y_i \neq y_j, \end{cases} \tag{7}$$

$$W^{(b)}_{i,j} \equiv \begin{cases} 1/n - 1/n_\ell & \text{if } y_i = y_j = \ell, \\ 1/n & \text{if } y_i \neq y_j. \end{cases} \tag{8}$$

A proof of Lemma 1 is given in Appendix A. Note that $1/n - 1/n_\ell$ in Eq. (8) is negative while $1/n_\ell$ and $1/n$ in Eqs. (7) and (8) are positive. This implies that if the data pairs in the same class are made close, the within-class scatter matrix $S^{(w)}$ gets 'small' and the between-class scatter matrix $S^{(b)}$ gets 'large'. On the other hand, if the data pairs in different classes are separated from each other, the between-class scatter matrix $S^{(b)}$ gets 'large'. Therefore, we may interpret FDA as keeping the sample pairs in the same class close and the sample pairs in different classes apart. A more formal discussion on the above interpretation is given in Appendix B.

3.2 Definition and Typical Behavior of LFDA

Based on the above pairwise expression, let us define the *local* within-class scatter matrix $\widetilde{S}^{(w)}$ and the *local* between-class scatter matrix $\widetilde{S}^{(b)}$ as follows.

$$\widetilde{S}^{(w)} \equiv \frac{1}{2}\sum_{i,j=1}^{n} \widetilde{W}^{(w)}_{i,j}(x_i - x_j)(x_i - x_j)^\top, \tag{9}$$

$$\widetilde{S}^{(b)} \equiv \frac{1}{2}\sum_{i,j=1}^{n} \widetilde{W}^{(b)}_{i,j}(x_i - x_j)(x_i - x_j)^\top,$$

where

$$\widetilde{W}^{(w)}_{i,j} \equiv \begin{cases} A_{i,j}/n_\ell & \text{if } y_i = y_j = \ell, \\ 0 & \text{if } y_i \neq y_j, \end{cases} \tag{10}$$

$$\widetilde{W}^{(b)}_{i,j} \equiv \begin{cases} A_{i,j}(1/n - 1/n_\ell) & \text{if } y_i = y_j = \ell, \\ 1/n & \text{if } y_i \neq y_j. \end{cases} \tag{11}$$

Namely, according to the affinity $A_{i,j}$, we weight the values for the sample pairs in the same class. This means that *far apart* sample pairs in the same class have less influence on $\widetilde{S}^{(w)}$ and $\widetilde{S}^{(b)}$. Note that we do *not* weight the values for the sample pairs in different classes since we want to separate them from each other *irrespective* of the affinity in the original space. From here on, we denote the local counterparts of matrices by symbols with *tilde*.

We define the LFDA transformation matrix T_{LFDA} as

$$T_{LFDA} \equiv \underset{T \in \mathbb{R}^{d \times r}}{\operatorname{argmax}} \left[\operatorname{tr} \left((T^\top \widetilde{S}^{(w)} T)^{-1} T^\top \widetilde{S}^{(b)} T \right) \right]. \tag{12}$$

That is, we look for a transformation matrix T such that *nearby* data pairs in the same class are made close and the data pairs in different classes are separated from each other; far apart data pairs in the same class are not imposed to be close.

Eq. (12) is of the same form as Eq. (3). Therefore, we can similarly compute an *analytic* form of T_{LFDA} by solving a generalized eigenvalue problem of $\widetilde{S}^{(b)}$ and $\widetilde{S}^{(w)}$. An efficient implementation of LFDA is summarized as a pseudo code in Figure 2 (see Appendix C for detail).

Toy examples of dimensionality reduction by LFDA are illustrated in Figure 1. We used the local scaling method for computing the affinity matrix A (see Appendix D.4). Note that we perform the nearest neighbor search in the local scaling method in a *classwise* manner since we do not need the affinity values for the sample pairs in different classes (see Eqs. 10 and 11). This highly contributes to reducing the computational cost (see Appendix C). Figure 1 shows that LFDA gives desirable results for all three data sets, that is, LFDA can compensate for the drawbacks of FDA and LPP by effectively combining the ideas of FDA and LPP.

If the affinity value $A_{i,j}$ is set to 1 for all sample pairs (i.e., all pairs are 'equally close' to each other), $\widetilde{S}^{(w)}$ and $\widetilde{S}^{(b)}$ agree with $S^{(w)}$ and $S^{(b)}$, respectively, and LFDA is reduced to the original FDA. Therefore, LFDA may be regarded as a natural localized variant of FDA.

3.3 Properties of LFDA

Here we discuss fundamental properties of LFDA.

First, we give an interpretation of LFDA in terms of the 'pointwise scatter'. $\widetilde{S}^{(w)}$ can be expressed as

$$\widetilde{S}^{(w)} = \frac{1}{2} \sum_{i=1}^{n} \frac{1}{n_{y_i}} \widetilde{P}_i^{(w)},$$

where n_{y_i} is the number of samples in the class to which the sample x_i belongs and $\widetilde{P}_i^{(w)}$ is the *pointwise* local within-class scatter matrix around x_i:

$$\widetilde{P}_i^{(w)} \equiv \sum_{j: y_j = y_i} A_{i,j} (x_j - x_i)(x_j - x_i)^\top.$$

Therefore, 'minimizing' $\widetilde{S}^{(w)}$ corresponds to minimizing the weighted sum of the pointwise local within-class scatter matrices over all samples. $\widetilde{S}^{(b)}$ can also be expressed in a similar way as

$$\widetilde{S}^{(b)} = \frac{1}{2} \sum_{i=1}^{n} \left(\frac{1}{n} - \frac{1}{n_{y_i}} \right) \widetilde{P}_i^{(w)} + \frac{1}{2n} \sum_{i=1}^{n} P_i^{(b)}, \tag{13}$$

where $P_i^{(b)}$ is the *pointwise* between-class scatter matrix around x_i:

$$P_i^{(b)} \equiv \sum_{j: y_j \neq y_i} (x_j - x_i)(x_j - x_i)^\top.$$

```
Input:   Labeled samples $\{(x_i, y_i) \mid x_i \in \mathbb{R}^d, y_i \in \{1, 2, \ldots, c\}\}_{i=1}^n$
         Dimensionality of embedding space $r$ $(1 \le r \le d)$
Output:  $d \times r$ transformation matrix $T_{LFDA}$

1:  $\widetilde{S}^{(b)} \longleftarrow 0_{d \times d}$;
2:  $\widetilde{S}^{(w)} \longleftarrow 0_{d \times d}$;
3:  for $\ell = 1, 2, \ldots, c$      % Compute scatter matrices in a classwise manner
4:      $\{\underline{x}_i\}_{i=1}^{n_\ell} \longleftarrow \{x_j\}_{j:y_j=\ell}$;
5:      for $i = 1, 2, \ldots, n_\ell$      % Determine local scaling
6:          $\underline{x}_i^{(7)} \longleftarrow$ 7th nearest neighbor of $\underline{x}_i$ among $\{\underline{x}_j\}_{j=1}^{n_\ell}$;
7:          $\underline{\sigma}_i \longleftarrow \|\underline{x}_i - \underline{x}_i^{(7)}\|$;
8:      end
9:      for $i, j = 1, 2, \ldots, n_\ell$      % Define affinity matrix
10:         $\underline{A}_{i,j} \longleftarrow \exp(-\|\underline{x}_i - \underline{x}_j\|^2 / (\underline{\sigma}_i \underline{\sigma}_j))$;
11:     end
12:     $\underline{X} \longleftarrow (\underline{x}_1 | \underline{x}_2 | \cdots | \underline{x}_{n_\ell})$;
13:     $\underline{G} \longleftarrow \underline{X}\mathrm{diag}(\underline{A}1_{n_\ell})\underline{X}^\top - \underline{X}\,\underline{A}\,\underline{X}^\top$;
14:     $\widetilde{S}^{(b)} \longleftarrow \widetilde{S}^{(b)} + \underline{G}/n + (1 - n_\ell/n)\underline{X}\,\underline{X}^\top + \underline{X}1_{n_\ell}(\underline{X}1_{n_\ell})^\top/n$;
15:     $\widetilde{S}^{(w)} \longleftarrow \widetilde{S}^{(w)} + \underline{G}/n_\ell$;
16: end
17: $\widetilde{S}^{(b)} \longleftarrow \widetilde{S}^{(b)} - X1_n(X1_n)^\top/n - \widetilde{S}^{(w)}$;
18: $\{\widetilde{\lambda}_k, \widetilde{\varphi}_k\}_{k=1}^r \longleftarrow$ generalized eigenvalues and normalized eigenvectors of
                $\widetilde{S}^{(b)}\widetilde{\varphi} = \widetilde{\lambda}\widetilde{S}^{(w)}\widetilde{\varphi}$;      % $\widetilde{\lambda}_1 \ge \widetilde{\lambda}_2 \ge \cdots \ge \widetilde{\lambda}_d$
19: $T_{LFDA} = (\sqrt{\widetilde{\lambda}_1}\widetilde{\varphi}_1 | \sqrt{\widetilde{\lambda}_2}\widetilde{\varphi}_2 | \cdots | \sqrt{\widetilde{\lambda}_r}\widetilde{\varphi}_r)$;
```

Figure 2: Efficient implementation of LFDA (see Appendix C for detail). The affinity matrix is computed by the local scaling method (see Appendix D.4). Matrices and vectors denoted with underline are classwise counterparts of the original ones. $0_{d \times d}$ denotes the $d \times d$ matrix with zeros, 1_{n_ℓ} denotes the n_ℓ-dimensional vector with ones, and $\mathrm{diag}(\underline{A}1_{n_\ell})$ denotes the diagonal matrix with diagonal elements $\underline{A}1_{n_\ell}$. The generalized eigenvectors in line 18 are normalized by Eq. (14), which is often automatically carried out by an eigensolver. The weighting scheme of the eigenvectors in line 19 is explained in Section 3.3. A possible bottleneck of the above implementation is the nearest neighbor search in line 6. This could be alleviated by incorporating the prior knowledge of the data structure or by approximation (see Saul and Roweis, 2003, and references therein). Another possible bottleneck is the computation of $\underline{X}\,\underline{A}\,\underline{X}^\top$ in line 13, which could be eased by sparsely defining the affinity matrix (see Appendix D). A MATLAB implementation is available from '`http://sugiyama-www.cs.titech.ac.jp/~sugi/software/LFDA/`'.

Note that $P_i^{(b)}$ does not include the localization factor $A_{i,j}$. Eq. (13) implies that 'maximizing' $\widetilde{S}^{(b)}$ corresponds to minimizing the weighted sum of the pointwise local within-class scatter matrices and maximizing the sum of the pointwise between-class scater matrices.

Next, we discuss the issue of eigenvalue multiplicity in LFDA. The original FDA allows us to extract at most $c-1$ meaningful features since the between-class scatter matrix $S^{(b)}$ has rank at most $c-1$ (Fukunaga, 1990). On the other hand, the local between-class scatter matrix $\widetilde{S}^{(b)}$ generally has a much higher rank with less eigenvalue multiplicity, thanks to the localization factor $A_{i,j}$ included in $\widetilde{W}^{(b)}$ (see Eq. 11). In the simulation shown in Section 5, $\widetilde{S}^{(b)}$ is always full rank for various data sets. Therefore, the proposed LFDA can be practically employed for dimensionality reduction into *any* dimensional spaces. This is a very important and significant improvement over the original FDA.

Finally, we discuss the invariance property of LFDA. The value of the LFDA criterion (12) is *invariant* under linear transformations, that is, for any r-dimensional invertible matrix H, $T_{LFDA}H$ is also a solution of Eq. (12). Therefore, the solution T_{LFDA} is not unique—the *range* of the transformation $H^\top T_{LFDA}^\top$ is uniquely determined, but the *distance metric* (Goldberger et al., 2005; Globerson and Roweis, 2006; Weinberger et al., 2006) in the embedding space can be arbitrary because of the arbitrariness of the matrix H. In practice, we propose determining the LFDA transformation matrix T_{LFDA} as follows. First, we rescale the generalized eigenvectors $\{\widetilde{\varphi}_k\}_{k=1}^d$ so that

$$\widetilde{\varphi}_k \widetilde{S}^{(w)} \widetilde{\varphi}_{k'} = \begin{cases} 1 & \text{if } k = k', \\ 0 & \text{if } k \neq k'. \end{cases} \tag{14}$$

Note that this rescaling is often automatically carried out by an eigensolver. Then we weight each generalized eigenvector by the square root of its associated generalized eigenvalue, that is,

$$T_{LFDA} = (\sqrt{\widetilde{\lambda}_1}\widetilde{\varphi}_1 | \sqrt{\widetilde{\lambda}_2}\widetilde{\varphi}_2 | \cdots | \sqrt{\widetilde{\lambda}_r}\widetilde{\varphi}_r), \tag{15}$$

where $\widetilde{\lambda}_1 \geq \widetilde{\lambda}_2 \geq \cdots \geq \widetilde{\lambda}_d$. This weighting scheme weakens the influence of minor eigenvectors and is shown to work well in experiments (see Section 5).

3.4 Kernel LFDA for Non-Linear Dimensionality Reduction

Here we show how LFDA can be extended to non-linear dimensionality reduction scenarios.

As detailed in Appendix C, the generalized eigenvalue problem that needs to be solved in LFDA can be expressed as

$$X\widetilde{L}^{(b)}X^\top \widetilde{\varphi} = \widetilde{\lambda} X \widetilde{L}^{(w)} X^\top \widetilde{\varphi}, \tag{16}$$

where $\widetilde{L}^{(b)} = \widetilde{L}^{(m)} - \widetilde{L}^{(w)}$ and $\widetilde{L}^{(m)}$ and $\widetilde{L}^{(w)}$ are defined by Eqs. (33) and (35), respectively. Since $X^\top \widetilde{\varphi}$ in Eq. (16) belongs to the range of $X^\top$, it can be expressed by using some vector $\widetilde{\alpha} \in \mathbb{R}^n$ as

$$X^\top \widetilde{\varphi} = X^\top X \widetilde{\alpha} = K\widetilde{\alpha},$$

where K is the n-dimensional matrix with the (i,j)-th element being

$$K_{i,j} \equiv x_i^\top x_j.$$

Then multiplying Eq. (16) by $X^\top$ from the left-hand side yields

$$K\widetilde{L}^{(b)}K\widetilde{\alpha} = \widetilde{\lambda}K\widetilde{L}^{(w)}K\widetilde{\alpha}. \tag{17}$$

This implies that $\{x_i\}_{i=1}^n$ appear only in terms of their *inner products*. Therefore, we can obtain a non-linear variant of LFDA by the *kernel trick* (Vapnik, 1998; Schölkopf et al., 1998), which is explained below.

Let us consider a non-linear mapping $\phi(x)$ from $\mathbb{R}^d$ to a *reproducing kernel Hilbert space* $\mathcal{H}$ (Aronszajn, 1950). Let $K(x,x')$ be the reproducing kernel of $\mathcal{H}$. A typical choice of the kernel function would be the Gaussian kernel:

$$K(x,x') = \exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right),$$

with $\sigma > 0$. For other choices, see, for example, Wahba (1990), Vapnik (1998), and Schölkopf and Smola (2002). Because of the reproducing property of $K(x,x')$, K is now the *kernel matrix*, that is, the (i,j)-th element is given by

$$K_{i,j} = \langle\phi(x_i),\phi(x_j)\rangle = K(x_i,x_j),$$

where $\langle\cdot,\cdot\rangle$ denotes the inner product in $\mathcal{H}$.

It can be confirmed that $\widetilde{L}^{(w)}$ is always degenerated (since $\widetilde{L}^{(w)}(1,1,\ldots,1)^\top$ always vanishes; see Eq. 35 for detail). Therefore, $K\widetilde{L}^{(w)}K$ is always degenerated and we cannot directly solve the generalized eigenvalue problem (17). To cope with this problem, we propose regularizing $K\widetilde{L}^{(w)}K$ and solving the following generalized eigenvalue problem instead (cf. Friedman, 1989).

$$K\widetilde{L}^{(b)}K\widetilde{\alpha} = \widetilde{\lambda}(K\widetilde{L}^{(w)}K + \varepsilon I_n)\widetilde{\alpha}, \tag{18}$$

where ε is a small constant. Let $\{\widetilde{\alpha}_k\}_{k=1}^n$ be the generalized eigenvectors associated with the generalized eigenvalues $\widetilde{\lambda}_1 \geq \widetilde{\lambda}_2 \geq \cdots \geq \widetilde{\lambda}_n$ of Eq. (18). Then the embedded image of $\phi(x')$ in $\mathcal{H}$ is given by

$$(\sqrt{\widetilde{\lambda}_1}\widetilde{\alpha}_1|\sqrt{\widetilde{\lambda}_2}\widetilde{\alpha}_2|\cdots|\sqrt{\widetilde{\lambda}_r}\widetilde{\alpha}_r)^\top \begin{pmatrix} K(x_1,x') \\ K(x_2,x') \\ \vdots \\ K(x_n,x') \end{pmatrix}.$$

We call this kernelized variant of LFDA *kernel LFDA* (KLFDA).

Recently, kernel functions for non-vectorial structured data such as strings, trees, and graphs have been proposed (see, e.g., Lodhi et al., 2002; Duffy and Collins, 2002; Kashima and Koyanagi, 2002; Kondor and Lafferty, 2002; Kashima et al., 2003; Gärtner et al., 2003; Gärtner, 2003). Since KLFDA uses the samples only via the kernel function $K(x,x')$, it allows us to reduce the dimensionality of such non-vectorial data.

4. Comparison with Related Methods

In this section, we discuss the relation between the proposed LFDA and other methods.

4.1 Dimensionality Reduction Using Local Discriminant Information

A *discriminant adaptive nearest neighbor* (DANN) classifier (Hastie and Tibshirani, 1996a) employs an adapted distance metric at each test point for classification. Based on a similar idea, they also proposed a global supervised dimensionality reduction method using *local discriminant information* (LDI) in the same paper. We refer to this supervised dimensionality reduction method as LDI. The main idea of LDI is to localize FDA—which is very similar to the proposed LFDA. Here we discuss the relation between LDI and LFDA.

In LDI, the data samples $\{x_i\}_{i=1}^n$ are first sphered according to the within-class scatter matrix $S^{(w)}$, that is, for $i = 1, 2, \ldots, n$,

$$\overline{x}_i \equiv (S^{(w)})^{-\frac{1}{2}} x_i.$$

Let $\overline{A}_{i,j}$ be the weight of sample $\overline{x}_j$ around $\overline{x}_i$ defined by

$$\overline{A}_{i,j} \equiv \begin{cases} \left[1 - \left(\frac{\|\overline{x}_i - \overline{x}_j\|}{\|\overline{x}_i - \overline{x}_i^{(K)}\|}\right)^3\right]^3 & \text{if } \|\overline{x}_i - \overline{x}_j\| < \|\overline{x}_i - \overline{x}_i^{(K)}\|, \\ 0 & \text{otherwise.} \end{cases}$$

where $\overline{x}_i^{(K)}$ is the K-th nearest neighbor of $\overline{x}_i$ in the sphered space. Note that $0 \leq \overline{A}_{i,j} \leq 1$ and $\overline{A}_{i,j}$ is non-increasing as $\|\overline{x}_i - \overline{x}_j\|$ increases. Thus it has the same meaning as our affinity matrix. K is suggested to be determined by

$$K = \max(n/5, 50).$$

Let $\overline{\mu}_\ell^{[i]}$ be the local weighted mean of the sphered samples in class ℓ around $\overline{x}_i$, and let $\overline{\mu}^{[i]}$ be the local weighted mean of the sphered samples around $\overline{x}_i$:

$$\overline{\mu}_\ell^{[i]} \equiv \frac{1}{\overline{n}_\ell^{[i]}} \sum_{j:y_j=\ell} \overline{A}_{i,j} \overline{x}_j,$$

$$\overline{\mu}^{[i]} \equiv \frac{1}{\overline{n}^{[i]}} \sum_{j=1}^{n} \overline{A}_{i,j} \overline{x}_j = \frac{1}{\overline{n}^{[i]}} \sum_{\ell=1}^{c} \overline{n}_\ell^{[i]} \overline{\mu}_\ell^{[i]},$$

where

$$\overline{n}_\ell^{[i]} \equiv \sum_{j:y_j=\ell} \overline{A}_{i,j},$$

$$\overline{n}^{[i]} \equiv \sum_{j=1}^{n} \overline{A}_{i,j}.$$

Let $\overline{S}^{(b)}$ be the *average between sum-of-squares matrix* defined as

$$\overline{S}^{(b)} \equiv \sum_{i=1}^{n} \frac{1}{\overline{n}^{[i]}} \sum_{\ell=1}^{c} \overline{n}_\ell^{[i]} (\overline{\mu}_\ell^{[i]} - \overline{\mu}^{[i]})(\overline{\mu}_\ell^{[i]} - \overline{\mu}^{[i]})^\top.$$

The LDI transformation matrix $\overline{T}_{LDI}$ is defined as

$$\overline{T}_{LDI} \equiv \underset{\overline{T} \in \mathbb{R}^{d \times r}}{\operatorname{argmax}} \left[\overline{T}^\top \overline{S}^{(b)} \overline{T}\right]$$

$$\text{subject to } \overline{T}^\top \overline{T} = I_r.$$

$\overline{T}_{LDI}$ is a transformation matrix for sphered samples; the LDI transformation matrix T_{LDI} for non-sphered samples is given by

$$T_{LDI} = (S^{(w)})^{-\frac{1}{2}}\overline{T}_{LDI}.$$

Similar to FDA (and LFDA), $\overline{T}_{LDI}$ can be efficiently computed by solving a generalized eigenvalue problem.

The average between sum-of-squares matrix $\overline{S}^{(b)}$ is conceptually very similar to the local between-class scatter matrix $\widetilde{S}^{(b)}$ in LFDA. Indeed, as proved in Appendix E, we can express $\overline{S}^{(b)}$ in a pairwise manner as

$$\overline{S}^{(b)} = \frac{1}{2}\sum_{i,j=1}^{n} \overline{W}^{(b)}_{i,j}(\overline{x}_i - \overline{x}_j)(\overline{x}_i - \overline{x}_j)^\top, \tag{19}$$

where

$$\overline{W}^{(b)}_{i,j} \equiv \begin{cases} \displaystyle\sum_{k=1}^{n} \frac{1}{\overline{n}^{[k]}}\left(\frac{1}{\overline{n}^{[k]}} - \frac{1}{\overline{n}^{[k]}_{\ell}}\right)\overline{A}_{i,k}\overline{A}_{j,k} & \text{if } y_i = y_j = \ell, \\ \displaystyle\sum_{k=1}^{n} \frac{1}{(\overline{n}^{[k]})^2}\overline{A}_{i,k}\overline{A}_{j,k} & \text{if } y_i \neq y_j. \end{cases} \tag{20}$$

However, there exist critical differences between LDI and LFDA. A significant difference is that the values for the sample pairs in different classes are also localized in LDI (see Eq. 20), while they are kept unlocalized in LFDA (see Eq. 11). This implies that far apart sample pairs in different classes could be made close in LDI, which is not desirable in supervised dimensionality reduction. Furthermore, the computation of $\overline{S}^{(b)}$ is slightly less efficient than $\widetilde{S}^{(b)}$ since $\overline{W}^{(b)}$ includes the summation over k.

Another important difference between LDI and LFDA is that the within-class scatter matrix $S^{(w)}$ is *not* localized in LDI. However, as we showed in Section 3.1, the within-class scatter matrix $S^{(w)}$ also accounts for collapsing the within-class multimodal structure (i.e., far apart sample pairs in the same class are made close). This phenomenon is experimentally confirmed in Section 5.2.

4.2 Mixture Discriminant Analysis

FDA can be interpreted as maximum likelihood estimation of Gaussian distributions with common covariance and different means for each class. Based on this view, Hastie and Tibshirani (1996b) proposed *mixture discriminant analysis* (MDA), which extends FDA to maximum likelihood estimation of Gaussian *mixture* distributions.

A maximum likelihood solution is obtained by an EM-type algorithm (cf. Dempster et al., 1977). However, this is an iterative algorithm and gives only a local optimal solution. Therefore, the computation of MDA is rather slow and there is no guarantee that the global solution can be obtained. Furthermore, the number of mixture components (clusters) in each class as well as the initial location of cluster centers should be determined by users. For cluster centers, using standard techniques such as k-means clustering (MacQueen, 1967; Everitt et al., 2001) or learning vector quantization (Kohonen, 1989) are recommended. However, they are also iterative algorithms and have no guarantee that the global solution can be obtained. Furthermore, there seems to be no systematic method for determining the number of clusters.

On the other hand, the proposed LFDA contains no tuning parameters (given that the affinity matrix is determined by the local scaling method, see Appendix D.4) and the global solution can be obtained analytically. However, it still lacks a probabilistic interpretation, which remains open currently.

4.3 Neighborhood Component Analysis

Goldberger et al. (2005) proposed a supervised dimensionality reduction method called *neighborhood component analysis* (NCA). The NCA transformation matrix T_{NCA} is defined as follows.

$$T_{NCA} \equiv \underset{T\in\mathbb{R}^{d\times r}}{\mathrm{argmax}} \left(\sum_{i=1}^{n} \sum_{j:y_j=y_i} p_{i,j}(TT^\top) \right), \tag{21}$$

where

$$p_{i,j}(U) \equiv \begin{cases} \dfrac{\exp\left\{-(x_i-x_j)^\top U(x_i-x_j)\right\}}{\sum_{k\neq i}\exp\left\{-(x_i-x_k)^\top U(x_i-x_k)\right\}} & \text{if } i\neq j, \\ 0 & \text{if } i=j. \end{cases} \tag{22}$$

The above definition corresponds to maximizing the expected number of correctly classified samples by a stochastic variant of nearest neighbor classifiers. Therefore, NCA seeks a transformation matrix T such that the between-class separability is maximized.

Eqs. (21) and (22) imply that nearby data pairs in the same class are made close, which is similar to the proposed LFDA. Indeed, the simulation results in Section 5.2 show that NCA tends to preserve the multimodal structure of the data very well. However, a crucial weakness of NCA is optimization: the optimization problem (21) is *non-convex*. Therefore, there is no guarantee that the globally optimal solution can be obtained. Goldberger et al. (2005) proposed using a gradient ascent method for optimization:

$$T \longleftarrow T + \varepsilon\nabla J_{NCA}(T), \tag{23}$$

where ε (>0) is the step size and the gradient $\nabla J_{NCA}(T)$ is given by

$$\nabla J_{NCA}(T) = 2T\sum_{i=1}^{n}\left(\left\{\sum_{j:y_j=y_i} p_{i,j}(TT^\top)\right\}\left\{\sum_{j=1}^{n} p_{i,j}(TT^\top)(x_i-x_j)(x_i-x_j)^\top\right\} - \sum_{j:y_j=y_i} p_{i,j}(TT^\top)(x_i-x_j)(x_i-x_j)^\top\right).$$

The gradient ascent iteration (23) is computationally rather inefficient. Also, the choice of the step size ε is troublesome. If the step size is small enough, the convergence to one of the local optima is guaranteed but such a choice makes the convergence very slow; on the other hand, if the step size is too large, gradient flows oscillate and proper convergence properties may not be guaranteed anymore. Furthermore, the choice of the termination condition in the iterative algorithm is often cumbersome in practice.

Because of the non-convexity of the optimization problem, the quality of the obtained solution depends on the *initialization* of the matrix T. A useful heuristic to alleviate the local optimum problem is to employ the FDA (or LFDA) result as an initial matrix for optimization (Goldberger et al., 2005). In the experiments in Section 5, using the LFDA result as an initial matrix appears to be better than the random initialization. However, the local optima problem still remains even with the above heuristic.

When a dimensionality reduction technique is applied to classification tasks, we often want to embed the data samples into spaces with several different dimensions—the best dimensionality is later chosen by, for example, cross-validation (Stone, 1974; Wahba, 1990). In such a scenario, NCA requires to optimize the transformation matrix *individually* for each dimensionality r of the embedding space. On the other hand, LFDA needs to compute the transformation matrix *only once* for the largest r; its sub-matrices become the optimal solutions for smaller dimensions. Therefore, LFDA is computationally more efficient than NCA in this scenario.

A simple MATLAB implementation of NCA is available.[4] We use this software in Section 5.

4.4 Maximally Collapsing Metric Learning

In order to overcome the computational problem of NCA, Globerson and Roweis (2006) proposed an alternative method called *maximally collapsing metric learning* (MCML).

Let $p^*_{i,j}$ be the 'ideal' value of $p_{i,j}(U)$ defined by Eq. (22):

$$p^*_{i,j} \propto \begin{cases} 1 & \text{if } y_i = y_j, \\ 0 & \text{if } y_i \neq y_j, \end{cases}$$

where $p^*_{i,j}$ is normalized so that

$$\sum_{j \neq i} p^*_{i,j} = 1.$$

$p^*_{i,j}$ can be attained if all samples in the same class collapse into a *single* point while samples in other classes are mapped to other locations. In reality, however, any U may not be able to attain $p_{i,j}(U) = p^*_{i,j}$ exactly; instead the optimal approximation to $p^*_{i,j}$ under the *Kullback-Leibler divergence* (Kullback and Leibler, 1951) is obtained. This is formally defined as

$$U_{MCML} \equiv \underset{U \in \mathbb{R}^{d \times d}}{\operatorname{argmin}} \left(\sum_{i,j=1}^{n} p^*_{i,j} \log \frac{p^*_{i,j}}{p_{i,j}(U)} \right) \quad \text{subject to } U \in PSD(r), \tag{24}$$

where $PSD(r)$ is the set of all positive semidefinite matrices of rank r (i.e., r eigenvalues are positive and others are zero). Once U_{MCML} is obtained, the MCML transformation matrix T_{MCML} is computed by

$$T_{MCML} = (\phi_1 | \phi_2 | \cdots | \phi_r), \tag{25}$$

where $\{\phi_k\}_{k=1}^{r}$ are the eigenvectors associated with the *positive* eigenvalues $\eta_1 \geq \eta_2 \geq \cdots \geq \eta_r > 0$ of the following eigenvalue problem:

$$U_{MCML}\phi = \eta\phi.$$

One of the motivations of MCML is to alleviate the difficulty of optimization in NCA. However, MCML still has a weakness in optimization: the optimization problem (24) is convex only when $r = d$, that is, the dimensionality is not reduced but only the *distance metric* of the original space is changed. This means that if $r < d$ (which is our primal focus in this paper), we may not be able to

4. Implementation available at '`http://www.cs.berkeley.edu/~fowlkes/software/nca/`'.

obtain the globally optimal solution. Globerson and Roweis (2006) proposed the following heuristic algorithm to *approximate* T_{MCML}.

First, the optimization problem (24) with $r = d$ is solved:

$$\widehat{U}_{MCML} \equiv \mathop{\mathrm{argmin}}_{U \in \mathbb{R}^{d\times d}} \left(\sum_{i,j=1}^{n} p^*_{i,j} \log \frac{p^*_{i,j}}{p_{i,j}(U)} \right) \quad \text{subject to } U \in PSD(d). \tag{26}$$

Although Eq. (26) is convex, an analytic form of the unique optimal solution $\widehat{U}_{MCML}$ is not known yet. Globerson and Roweis (2006) proposed using the following alternate iterative procedure for obtaining $\widehat{U}_{MCML}$.

$$U \leftarrow U - \varepsilon \nabla J_{MCML}(U), \tag{27}$$

$$U \leftarrow \sum_{k=1}^{d} \max(0, \widehat{\eta}_k) \widehat{\varphi}_k \widehat{\varphi}_k^\top, \tag{28}$$

where ε (> 0) is the step size, $\widehat{\eta}_k$ and $\widehat{\varphi}_k$ are eigenvalues and eigenvectors of U, and the gradient $\nabla J_{MCML}(U)$ is given by

$$\nabla J_{MCML}(U) = \sum_{i,j=1}^{n} (p^*_{i,j} - p_{i,j}(U))(x_i - x_j)(x_i - x_j)^\top.$$

Then the eigenvalue decomposition of $\widehat{U}_{MCML}$ is carried out and eigenvalues $\widehat{\eta}_1 \geq \widehat{\eta}_2 \geq \cdots \geq \widehat{\eta}_d$ and associated eigenvectors $\{\widehat{\varphi}_k\}_{k=1}^{d}$ are obtained:

$$\widehat{U}_{MCML}\widehat{\varphi} = \widehat{\eta}\widehat{\varphi}.$$

Finally, $\{\varphi_k\}_{k=1}^{r}$ in Eq. (25) are replaced by $\{\widehat{\varphi}_k\}_{k=1}^{r}$, which yields

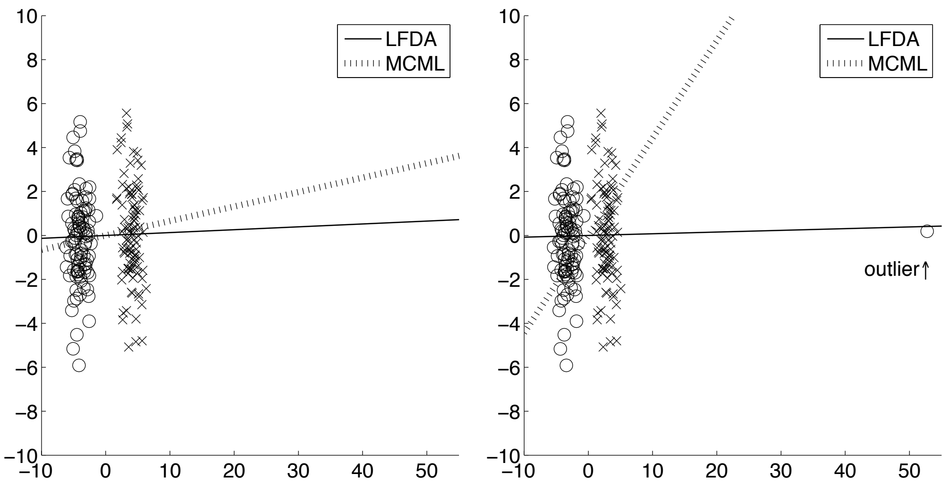
$$T_{MCML} \approx (\widehat{\varphi}_1 | \widehat{\varphi}_2 | \cdots | \widehat{\varphi}_r). \tag{29}$$

This approximation is shown to be practically useful (Globerson and Roweis, 2006), although there seems to be no theoretical analysis for this approximation.

MCML may have an advantage over NCA in computation: there exists the analytic approximation (29) that can be computed efficiently using the solution of another convex optimization problem (26). However, MCML still relies on the gradient-based alternate iterative algorithm (27)–(28) to solve the convex optimization problem (26), which is computationally very expensive since the eigenvalue decomposition of a d-dimensional matrix should be carried out in each iteration (see Eq. 28). Furthermore, the difficulty of appropriately choosing the step size and the termination condition in the iterative procedure still remains.

Since MCML requires all the samples in the same class to collapse into a single point, it is not necessarily useful in dimensionality reduction of multimodal data samples. Furthermore, the MCML results can be significantly influenced by outliers since the outliers are also required to collapse into the same single point together with other samples. This phenomenon is illustrated in Figure 3, where a single outlier significantly changes the MCML result.

Globerson and Roweis (2006) showed that the sufficient statistics of the MCML algorithm are pointwise scatter matrices (cf. Section 3.3). Since LFDA also has an interpretation in terms of pointwise scatter matrices, there may be a link between LFDA and MCML and this needs to be investigated in the future work.



(a) Toy data set 1

(b) Toy data set 1'

Figure 3: Toy examples of dimensionality reduction. The toy data set 1 is equivalent to the one used in Figure 1(a). The data set 1' includes a single outlier.

4.5 Remark on Rank Constraint

The optimization problem of MCML (see Eq. 24) is not generally convex since the rank constraint is non-convex (Boyd and Vandenberghe, 2004). The non-convexity induced by the rank constraint seems to be a universal problem in dimensionality reduction. NCA eliminates the rank constraint by decomposing U into $TT^\top$ (see Eqs. 21 and 22). However, even with this decomposition, the optimization problem is still non-convex. On the other hand, FDA, LDI, and LFDA cast the optimization problem in the form of the *Rayleigh quotient*. This is computationally very advantageous since it allows us to analytically determine the range of the embedding space. However, we cannot determine the distance metric in the embedding space since the Rayleigh quotient is invariant under linear transformations. For this reason, an additional criterion is needed to determine the distance metric (see also Section 3.3).

5. Numerical Examples

In this section, we numerically evaluate the performance of LFDA and existing methods.

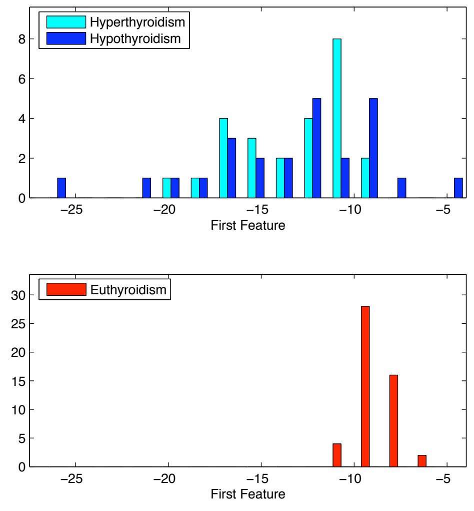
5.1 Exploratory Data Analysis

Here we use the *Thyroid disease* data set available from the UCI machine learning repository (Blake and Merz, 1998) and illustrate how LFDA can be used for exploratory data analysis.

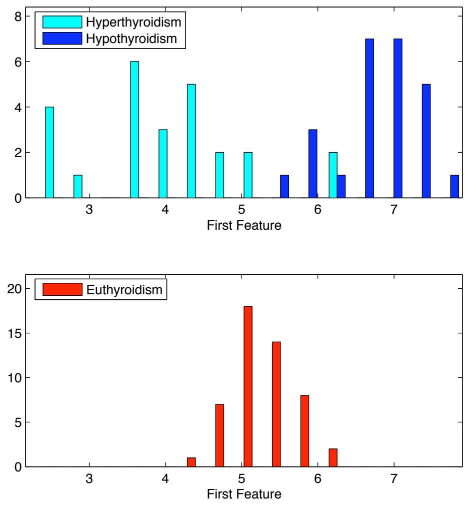
The original data consists of 5-dimensional input vector x of the following laboratory tests.

1. T3-resin uptake test.

2. Total Serum thyroxin as measured by the isotopic displacement method.



(a) FDA



(b) LFDA

Figure 4: Histograms of the first feature values obtained by FDA and LFDA for the *Thyroid disease* data set. The top row corresponds to the sick patients and the bottom row corresponds to the healthy patients.

3. Total Serum triiodothyronine as measured by radioimmuno assay.

4. Basal thyroid-stimulating hormone (TSH) as measured by radioimmuno assay.

5. Maximal absolute difference of TSH value after injection of 200 micro grams of thyrotropin-releasing hormone as compared to the basal value.

The task is to predict whether patients' thyroids are *euthyroidism*, *hypothyroidism*, or *hyperthyroidism* (Coomans et al., 1983), that is, whether patients' thyroids are normal, hypo-functioning, or hyper-functioning (Blake and Merz, 1998). The diagnosis (the class label) is based on a complete medical record, including anamnesis, scan etc. Here we merge the hypothyroidism class and the hyperthyroidism class into a single class and create binary labeled data (whether thyroids are normal or not). Our goal is to predict whether patients' thyroids are normal, hypo-functioning, or hyper-functioning from the binary labeled data samples.

Figure 4 depicts the histograms of the first feature values obtained by FDA and LFDA—the top row corresponds to the sick patients and the bottom row corresponds to the healthy patients. This shows that both FDA and LFDA separate the patients with normal thyroids from sick patients reasonably well. In addition to between-class separability, LFDA clearly preserves the multimodal structure among sick patients (i.e., hypo-functioning and hyper-functioning), which is lost by ordinary FDA. Another interesting finding from the figure is that the first feature values obtained by LFDA has a strong negative correlation to the functioning level of thyroids—this could be used for predicting the functioning level of thyroids.

Data Set	d	'$\circ$'-and-'$\bullet$' class	'$\times$' class
Letter recognition	16	'A' & 'C'	'B'
Iris	4	'Setosa' & 'Virginica'	'Versicolour'

Table 1: Two-class data sets used for visualization experiments ($r = 2$).

5.2 Data Visualization

Here we apply the proposed and existing dimensionality reduction methods to benchmark data sets and investigate how they behave in data visualization tasks.

We use the *Letter recognition* data set and the *Iris* data set available from the UCI machine learning repository (Blake and Merz, 1998). Table 1 describes the specifications of the data sets. Each data set contains three types of samples specified by '$\circ$', '$\bullet$', and '$\times$'. We merged '$\circ$' and '$\bullet$' into a single class and created two-class problems. We test LFDA, FDA, LPP, LDI, NCA, and MCML and evaluate the between-class separability (i.e., '$\circ$' and '$\bullet$' are well separated from '$\times$') and the within-class multimodality preservation capability (i.e., '$\circ$' and '$\bullet$' are well grouped). For LPP and LFDA, we determined the affinity matrix by the local scaling method (see Appendix D.4). For NCA, we used the LFDA result as an initial matrix since this initialization scheme appears to work better than the random initialization. FDA allows us to extract only one meaningful feature in two-class classification problems (see Section 2.2), so we choose the second feature randomly here. Figures 5 and 6 depict the samples embedded in the two-dimensional space found by each method. The horizontal axis is the first feature found by each method, while the vertical axis is the second feature.

First, we compare the embedding results of LFDA with those of FDA and LPP. For the *Letter recognition* data set (see the top row of Figure 5), LFDA nicely separates samples in different classes from each other, and at the same time, it clearly preserves within-class multimodality. FDA separates '$\circ$' and '$\bullet$' from '$\times$' well, but within-class multimodality is lost, that is, '$\circ$' and '$\bullet$' are mixed. LPP gives two separate clusters of samples, but samples in different classes are mixed in one of the clusters. For the *Iris* data set (see the top row of Figure 6), LFDA simultaneously achieves between-class separation and within-class multimodality preservation. On the other hand, FDA tends to mix samples in different classes, which would be caused by within-class multimodality. LPP also works well for this data set because three clusters are well separated from each other in the original high-dimensional space. Overall, LFDA is found to be more appropriate for embedding labeled multimodal data samples than FDA and LPP, implying that our primal goal has been successfully achieved.

Next, we compare the results of LFDA with those of LDI, NCA, and MCML. For the *Letter recognition* data set (see Figure 5), LFDA, LDI, NCA, and MCML separate the samples in different classes from each other very well. However, LDI and MCML collapse '$\circ$' and '$\bullet$' into a single cluster, while LFDA and NCA preserve the multimodal structure clearly. The NCA result is almost identical to the LFDA result (i.e., the initial value of the NCA iteration), but the result may vary if the initial value for the gradient ascent algorithm is changed. For the *Iris* data set (see Figure 6), LFDA, LDI, and NCA work excellently in both between-class separation and within-class multimodality preservation. On the other hand, MCML mixes the samples in different classes. Overall, LDI works fairly well, but the within-class multimodal structure is sometimes lost since LDI only partially takes within-class multimodality into account (see Section 4.1). NCA also works very well, which

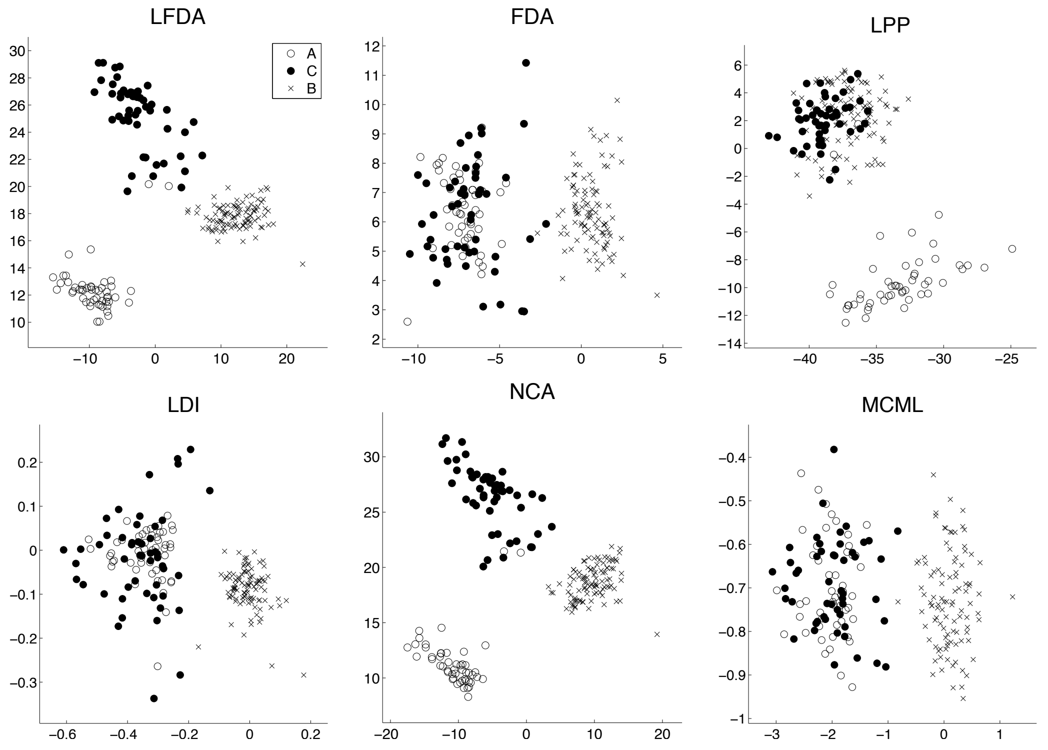


Figure 5: Visualization of the *Letter recognition* data set.

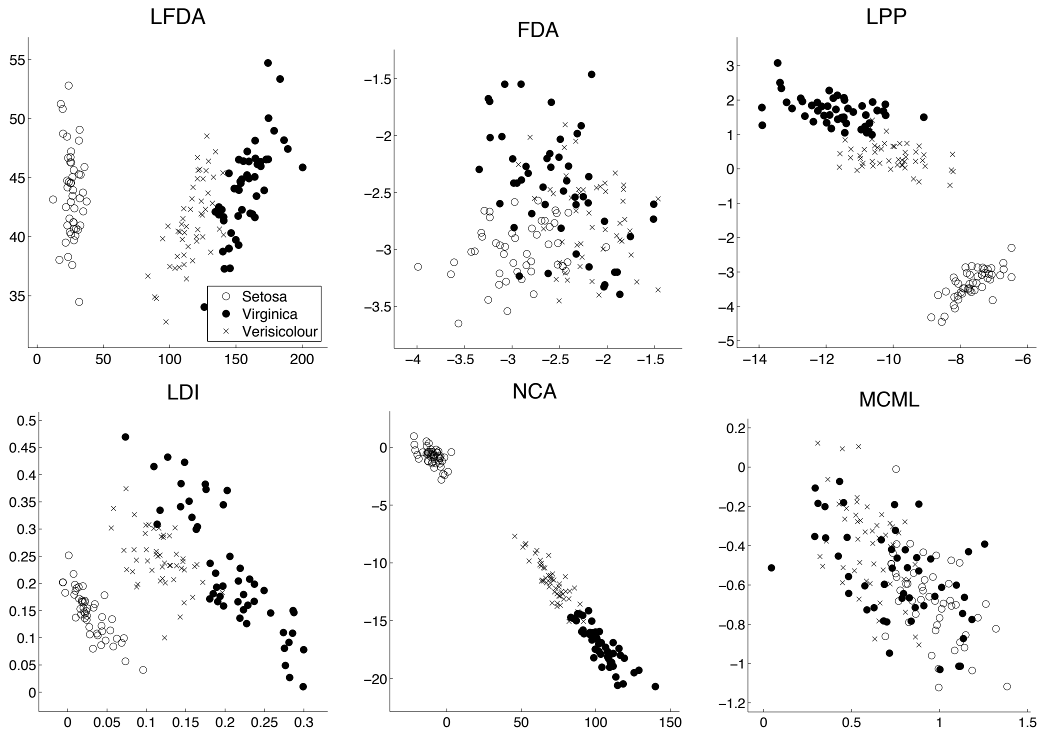


Figure 6: Visualization of the *Iris* data set.

Data name	Input dimensionality	# of training samples	# of test samples	# of realizations
*banana	2	400	4900	100
breast-cancer	9	200	77	100
diabetes	8	468	300	100
flare-solar	9	666	400	100
german	20	700	300	100
heart	13	170	100	100
image	18	1300	1010	20
ringnorm	20	400	7000	100
splice	60	1000	2175	20
*thyroid	5	140	75	100
titanic	3	150	2051	100
twonorm	20	400	7000	100
*waveform	21	400	4600	100
*USPS-eo	256	1000	1000	20
*USPS-sl	256	1000	1000	20

Table 2: List of binary classification data sets. Data sets indicated by '$*$' contain intrinsic within-class multimodal structures.

implies that the heuristic to use the LFDA result as an initial value is useful. However, NCA does not provide significant performance improvement over LFDA in the above simulations. The MCML results have similar tendencies to FDA.

Based on the above simulation results, we conclude that LFDA is a promising method in the visualization of multimodal labeled data.

5.3 Classification

Here we apply the proposed and existing dimensionality reduction techniques to classification tasks, and objectively evaluate the effectiveness of LFDA.

There are several measures for quantitatively evaluating separability of data samples in different classes (e.g., Fukunaga, 1990; Globerson et al., 2005). Here we use a simple one: *misclassification rate by a one-nearest-neighbor classifier*. As explained in Section 3.3, the LFDA criterion is invariant under linear transformations, while the misclassification rate by a one-nearest-neighbor classifier depends on the distance metric. This means that the following simulation results are highly dependent on the normalization scheme (15).

We employ the *IDA* data sets,[5] which are standard binary classification data sets originally used in Rätsch et al. (2001). In addition, we use two binary classification data sets created from the *USPS handwritten digit* data set. The first task (USPS-eo) is to separate even numbers from odd numbers and the second task (USPS-sl) is to separate small numbers ('0' to '4') from large numbers ('5' to '9'). For training and testing, 100 samples are randomly chosen for each digit. Table 2 summarizes

5. Data sets available at `http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm`.

Data set	LFDA	LDI	NCA	MCML	LPP	PCA
*banana	°13.7 ± 0.8	°13.6 ± 0.8	14.3 ± 2.0	39.4 ± 6.7	°13.6 ± 0.8	°13.6 ± 0.8
breast-cancer	°34.7 ± 4.3	36.4 ± 4.9	34.9 ± 5.0	°34.0 ± 5.8	°33.5 ± 5.4	°34.5 ± 5.0
diabetes	32.0 ± 2.5	°30.8 ± 1.9	—	°31.2 ± 2.1	31.5 ± 2.5	°31.2 ± 3.0
flare-solar	°39.2 ± 5.0	°39.3 ± 4.8	—	—	°39.2 ± 4.9	°39.1 ± 5.1
german	°29.9 ± 2.8	30.7 ± 2.4	°29.8 ± 2.6	31.3 ± 2.4	30.7 ± 2.4	°30.2 ± 2.4
heart	°21.9 ± 3.7	23.9 ± 3.1	23.0 ± 4.3	23.3 ± 3.8	23.3 ± 3.8	24.3 ± 3.5
image	°3.2 ± 0.8	°3.0 ± 0.6	—	4.7 ± 0.8	3.6 ± 0.7	°3.4 ± 0.5
ringnorm	21.1 ± 1.3	°17.5 ± 1.0	21.8 ± 1.3	22.0 ± 1.2	20.6 ± 1.1	21.6 ± 1.4
splice	°16.9 ± 0.9	17.9 ± 0.8	—	°17.3 ± 0.9	23.2 ± 1.2	22.6 ± 1.3
*thyroid	°4.6 ± 2.6	8.0 ± 2.9	°4.5 ± 2.2	18.5 ± 3.8	°4.2 ± 2.9	°4.9 ± 2.6
titanic	°33.1 ± 11.9	°33.1 ± 11.9	°33.0 ± 11.9	°33.1 ± 11.9	°33.0 ± 11.9	°33.0 ± 12.0
twonorm	°3.5 ± 0.4	4.1 ± 0.6	3.7 ± 0.6	°3.5 ± 0.4	3.7 ± 0.7	3.6 ± 0.6
*waveform	°12.5 ± 1.0	20.7 ± 2.5	°12.6 ± 0.8	17.9 ± 1.5	°12.4 ± 1.0	12.7 ± 1.2
*USPS1	9.0 ± 0.8	12.5 ± 0.9	—	—	7.2 ± 1.0	°3.5 ± 0.7
*USPS2	12.9 ± 1.2	25.9 ± 1.7	—	11.7 ± 1.3	7.5 ± 0.8	°3.9 ± 0.8
Computation time (ratio)	1.00	1.11	97.23	70.61	1.04	0.91

Table 3: Means and standard deviations of the misclassification rate when the embedding dimensionality is chosen by cross validation. For each data set, the best method and comparable ones based on the t-test at the significance level 5% are marked by '∘'. Data sets indicated by '∗' contain the intrinsic within-class multimodal structure.

the specifications of the data sets. The *ringnorm*, *twonorm*, and *waveform* data sets contain features with only noise. The *thyroid*, *waveform*, *USPS-eo*, and *USPS-sl* data sets contain intrinsic within-class multimodal structures since they are converted from multi-class problems by merging some of the classes. The *banana* data set is also multimodal.

We test LFDA, LDI, NCA, MCML, LPP, and principal component analysis (PCA). Note that LPP and PCA are unsupervised dimensionality reduction methods, while others are supervised methods. NCA is not tested for the *diabetes*, *flare-solar*, *image*, *splice*, *USPS-eo*, and *USPS-sl* data sets and MCML is not tested for the *flare-solar* and *USPS-eo* data sets since the execution time is too long.

Figure 7 depicts the mean misclassification rate by a one-nearest-neighbor classifier as functions of the dimensionality r of the reduced space. The error bars are omitted for clear visibility. Instead, we plotted the results of the following significance test: for each dimensionality r, the mean misclassification rate by the best method and comparable ones based on the *t-test* (Henkel, 1979) at the significance level 5% are marked by '∘'. The results show that LFDA works quite well, but overall there is no single best method that consistently outperforms the others.

Table 3 describes the mean and standard deviation of the misclassification rate by each method when the embedding dimensionality r is chosen by 5-*fold cross validation* (Stone, 1974; Wahba, 1990); for the *USPS-eo* and *USPS-sl* data sets, we used 20-*fold cross validation* since this was more accurate. For each data set, the best method and comparable ones based on the t-test at the significance level 5% are indicated by '∘'. The table shows that overall LFDA has excellent

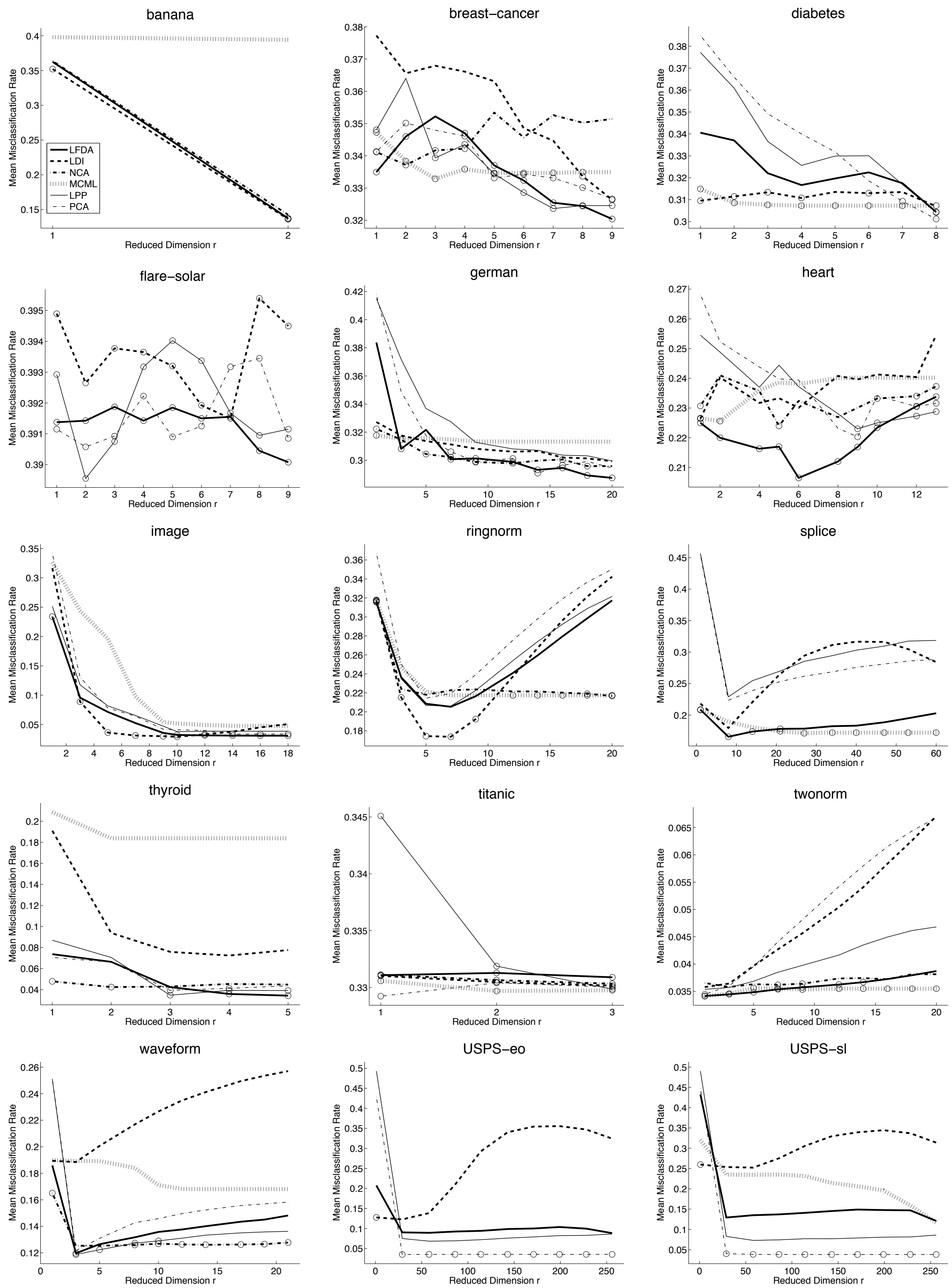


Figure 7: Mean misclassification rates by a one-nearest-neighbor method as functions of the dimensionality of the embedding space. For each dimension, the best method and comparable ones based on the t-test at the significance level 5% are marked by '○'.

Data set	LFDA	EUCLID	FDA
*banana	$^\circ 13.7 \pm 0.8$	$^\circ 13.6 \pm 0.8$	$^\circ 13.6 \pm 0.8$
breast-cancer	34.7 ± 4.3	$^\circ 32.7 \pm 4.8$	$^\circ 32.9 \pm 4.5$
diabetes	32.0 ± 2.5	$^\circ 30.1 \pm 2.1$	$^\circ 30.6 \pm 2.2$
flare-solar	$^\circ 39.2 \pm 5.0$	$^\circ 39.2 \pm 5.1$	$^\circ 39.0 \pm 4.9$
german	$^\circ 29.9 \pm 2.8$	$^\circ 29.5 \pm 2.5$	30.5 ± 2.8
heart	$^\circ 21.9 \pm 3.7$	23.2 ± 3.7	24.0 ± 3.7
image	$^\circ 3.2 \pm 0.8$	$^\circ 3.4 \pm 0.5$	6.5 ± 1.7
ringnorm	$^\circ 21.1 \pm 1.3$	35.0 ± 1.4	31.2 ± 1.6
splice	$^\circ 16.9 \pm 0.9$	28.9 ± 1.5	33.7 ± 1.5
*thyroid	$^\circ 4.6 \pm 2.6$	$^\circ 4.4 \pm 2.2$	5.3 ± 2.5
titanic	$^\circ 33.1 \pm 11.9$	$^\circ 33.0 \pm 11.9$	$^\circ 33.1 \pm 11.9$
twonorm	$^\circ 3.5 \pm 0.4$	6.7 ± 0.7	5.0 ± 0.7
*waveform	$^\circ 12.5 \pm 1.0$	15.8 ± 0.7	17.6 ± 1.4
*USPS-eo	9.0 ± 0.8	$^\circ 3.6 \pm 0.7$	15.1 ± 2.4
*USPS-sl	12.9 ± 1.2	$^\circ 3.8 \pm 0.8$	13.3 ± 2.9

Table 4: Means and standard deviations of the misclassification rate. The LFDA results are copied from Table 3. 'EUCLID' denotes a naive one-nearest-neighbor classification without dimensionality reduction. 'FDA' denotes a naive one-nearest-neighbor classification after the samples are projected onto a one-dimensional FDA subspace.

performance. LDI and MCML also work quite well, but they tend to perform rather poorly for the multimodal data sets specified by '$*$'. NCA also works well, but it does not compare favorably with LFDA. Note that NCA with random initialization was slightly worse; therefore our heuristic to use the LFDA results for initialization would be reasonable. LPP and PCA perform well, despite the fact that they are unsupervised dimensionality reduction methods. In particular, PCA has excellent performance for the USPS data sets since the projection onto the two-dimensional PCA subspace already gives reasonably separate embedding (He and Niyogi, 2004).

The computation time of each method summed over 9 data sets for which NCA is tested is described in the bottom of Table 3. For better comparison, we normalized the values by the computation time of LFDA. This shows that LFDA is much faster than NCA and MCML,[6] and is comparable to LDI, LPP, and PCA.

The misclassification rates by a naive one-nearest-neighbor classification without dimensionality reduction ('EUCLID') are described in Table 4. The table shows that, on the whole, the performance of LFDA is comparable to EUCLID. This implies that the use of LFDA is advantageous when the dimensionality of the original data is very high since the computation time in the test phase can be reduced. Table 4 also includes the misclassification rates by a naive one-nearest-neighbor classification after the samples are projected onto a one-dimensional FDA subspace, showing that LFDA tends to outperform FDA.

6. In our implementation of MCML, we used a constant step size for the gradient descent. The computation time could be improved if, for example, an Armijo like step size rule (Bertsekas, 1976) is employed.

Based on the above simulation results, we conclude that the proposed LFDA is a promising dimensionality reduction technique also in classification scenarios.

6. Conclusions

We discussed the problem of supervised dimensionality reduction. FDA (Fisher, 1936; Fukunaga, 1990; Duda et al., 2001) works well for this purpose, given that data samples in each class are *unimodal* Gaussian. However, samples in a class are often multimodal in practice, for example, when multi-class classification problems are solved by a set of two-class 'one-versus-rest' problems. LPP (He and Niyogi, 2004) can work well in dimensionality reduction of multimodal data. However, it is an unsupervised method and does not necessarily useful in supervised dimensionality reduction scenarios. In this paper, we proposed a new method called LFDA, which effectively combines the ideas of FDA and LPP. LFDA allows us to reduce the dimensionality of multimodal labeled data appropriately by maximizing between-class separability and preserving the within-class *local* structure at the same time. The derivation of LFDA is based on a novel *pairwise* interpretation of FDA (see Section 3.1). The original FDA provides a meaningful result only when the dimensionality of the embedding space is smaller than the number of classes because of the rank deficiency of the between-class scatter matrix. On the other hand, LFDA does not share this limitation and can be employed for dimensionality reduction into *any* dimensional spaces (see Section 3.3). This is a significant improvement over the original FDA.

As discussed in Section 3.3, the LFDA criterion is invariant under linear transformations. This means that the *range* of the transformation matrix can be uniquely determined, but the *distance metric* in the embedding space cannot be determined. In this paper, we determined the distance metric in a heuristic manner. Although this normalization scheme is shown to be reasonable in experiments, there is still room for further improvement. An important future direction is to develop a computationally efficient method of determining the distance metric of the embedding space, for example, following the lines of Goldberger et al. (2005), Globerson and Roweis (2006), and Weinberger et al. (2006).

We showed in Section 3.4 that a non-linear variant of LFDA can be obtained by employing the kernel trick. FDA, LPP, and MCML can also be kernelized similarly (Baudat and Anouar, 2000; Mika et al., 2003; Belkin and Niyogi, 2003; He and Niyogi, 2004; Globerson and Roweis, 2006). As shown in these papers, the performance of the kernelized methods heavily depend on the choice of the family and parameters of kernel functions. Therefore, how to optimally determine the kernel function for supervised dimensionality reduction needs to be explored.

The performance of LFDA depends on the choice of the affinity matrix. In this paper, we simply employed a standard definition as it is (see Appendix D.4). Although this standard choice appeared to be reasonable in experiments, it is important to find the optimal way to define the affinity matrix in the context of supervised dimensionality reduction.

MDA (Hastie and Tibshirani, 1996b) provides a solid probabilistic framework for supervised dimensionality reduction with multimodality (see Section 4.2). On the other hand, LFDA still lacks a probabilistic interpretation. An interesting future direction is to analyze the behavior of LFDA in terms of density models.

Acknowledgments

The author would like to thank Klaus-Robert Müller, Hideki Asoh, Motoaki Kawanabe, and Stefan Harmeling for fruitful discussions. His gratitude also goes to anonymous reviewers for their valuable comments; particularly, the weighting scheme of the eigenvectors in Eq. (15) was suggested by one of the reviewers. He also acknowledges financial support from MEXT (Grant-in-Aid for Young Scientists 17700142 and Grant-in-Aid for Scientific Research (B) 18300057). A part of this work has been done when the author was staying at University of Edinburgh, which is supported by EU Erasmus Mundus Scholarship. He would like to thank Sethu Vijayakumar for the warm hospitality.

Appendix A. Proof of Lemma 1

It follows from Eq. (1) that

$$\begin{aligned}
S^{(w)} &= \sum_{\ell=1}^{c} \sum_{i:y_i=\ell} \left(x_i - \frac{1}{n_\ell} \sum_{j:y_j=\ell} x_j \right) \left(x_i - \frac{1}{n_\ell} \sum_{j:y_j=\ell} x_j \right)^\top \\
&= \sum_{i=1}^{n} x_i x_i^\top - \sum_{\ell=1}^{c} \frac{1}{n_\ell} \sum_{i,j:y_i=y_j=\ell} x_i x_j^\top \\
&= \sum_{i=1}^{n} \left(\sum_{j=1}^{n} W_{i,j}^{(w)} \right) x_i x_i^\top - \sum_{i,j=1}^{n} W_{i,j}^{(w)} x_i x_j^\top \\
&= \frac{1}{2} \sum_{i,j=1}^{n} W_{i,j}^{(w)} (x_i x_i^\top + x_j x_j^\top - x_i x_j^\top - x_j x_i^\top),
\end{aligned}$$

which yields Eq. (5). Let $S^{(m)}$ be the *mixture scatter matrix* (Fukunaga, 1990):

$$\begin{aligned}
S^{(m)} &\equiv S^{(w)} + S^{(b)} \\
&= \sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^\top.
\end{aligned}$$

Then we have

$$\begin{aligned}
S^{(b)} &= \sum_{i=1}^{n} x_i x_i^\top - \frac{1}{n} \sum_{i,j=1}^{n} x_i x_j^\top - S^{(w)} \\
&= \sum_{i=1}^{n} \left(\sum_{j=1}^{n} \frac{1}{n} \right) x_i x_i^\top - \sum_{i,j=1}^{n} \frac{1}{n} x_i x_j^\top - S^{(w)} \\
&= \frac{1}{2} \sum_{i,j=1}^{n} \left(\frac{1}{n} - W_{i,j}^{(w)} \right) (x_i x_i^\top + x_j x_j^\top - x_i x_j^\top - x_j x_i^\top),
\end{aligned}$$

which yields Eq. (6). ■

Appendix B. Interpretation of FDA

In Section 3.1, we claimed that FDA tries to keep data pairs in the same class 'close' and data pairs in different classes 'apart'. Here we show this claim more formally.

For

$$v_{i,j} \equiv T^\top (x_i - x_j),$$

let us investigate the change in the Fisher criterion (3) when $v_{i,j}$ yields $\alpha v_{i,j}$ with $\alpha > 0$. Note that there does not generally exist a transformation T' that keeps all $v_{i,j}$ and only changes a particular pair. For this reason, the following analysis may be regarded as comparing the values of the Fisher criterion for two *different* data sets. This analysis will give an insight into what kind of transformation matrices the Fisher criterion favors.

Let

$$\begin{aligned}
W &\equiv T^\top S^{(w)} T, \\
B &\equiv T^\top S^{(b)} T, \\
W_\alpha &\equiv W - \beta W_{i,j}^{(w)} v_{i,j} v_{i,j}^\top, \\
B_\alpha &\equiv B - \beta W_{i,j}^{(b)} v_{i,j} v_{i,j}^\top, \\
\beta &\equiv \frac{1-\alpha^2}{2}.
\end{aligned}$$

Note that W_α and B_α correspond to the within-class and between-class scatter matrices for $\alpha v_{i,j}$, respectively. We assume that W and W_α are positive definite and B and B_α are positive semi-definite. Then the values of the Fisher criterion (3) for $v_{i,j}$ and $\alpha v_{i,j}$ are expressed as $\mathrm{tr}\left(W^{-1}B\right)$ and $\mathrm{tr}\left(W_\alpha^{-1}B_\alpha\right)$, respectively.

The standard *matrix inversion lemma* (e.g., Albert, 1972) yields

$$\begin{aligned}
W^{-1} &= (W_\alpha + \beta W_{i,j}^{(w)} v_{i,j} v_{i,j}^\top)^{-1} \\
&= W_\alpha^{-1} - \frac{W_\alpha^{-1} v_{i,j} (W_\alpha^{-1} v_{i,j})^\top}{(\beta W_{i,j}^{(w)})^{-1} + \langle W_\alpha^{-1} v_{i,j}, v_{i,j} \rangle}.
\end{aligned}$$

If $y_i = y_j$, we have $W_{i,j}^{(w)} > 0$ and $W_{i,j}^{(b)} < 0$. Then we have

$$\begin{aligned}
\mathrm{tr}\left(W^{-1}B\right) &= \mathrm{tr}\left((W_\alpha + \beta W_{i,j}^{(w)} v_{i,j} v_{i,j}^\top)^{-1} (B_\alpha + \beta W_{i,j}^{(b)} v_{i,j} v_{i,j}^\top)\right) \\
&= \mathrm{tr}\left(W_\alpha^{-1} B_\alpha\right) + \beta W_{i,j}^{(b)} \langle W_\alpha^{-1} v_{i,j}, v_{i,j} \rangle \\
&\quad - \frac{\langle B_\alpha W_\alpha^{-1} v_{i,j}, W_\alpha^{-1} v_{i,j} \rangle + \beta W_{i,j}^{(b)} \langle W_\alpha^{-1} v_{i,j}, v_{i,j} \rangle^2}{(\beta W_{i,j}^{(w)})^{-1} + \langle W_\alpha^{-1} v_{i,j}, v_{i,j} \rangle} \\
&= \mathrm{tr}\left(W_\alpha^{-1} B_\alpha\right) - \frac{\langle B_\alpha W_\alpha^{-1} v_{i,j}, W_\alpha^{-1} v_{i,j} \rangle - W_{i,j}^{(b)} {W_{i,j}^{(w)}}^{-1} \langle W_\alpha^{-1} v_{i,j}, v_{i,j} \rangle}{(\beta W_{i,j}^{(w)})^{-1} + \langle W_\alpha^{-1} v_{i,j}, v_{i,j} \rangle}.
\end{aligned} \tag{30}$$

If $\alpha < 1$, we have $\beta > 0$ since $\alpha > 0$ by definition. Therefore, Eq. (30) yields

$$\mathrm{tr}\left(W_\alpha^{-1} B_\alpha\right) > \mathrm{tr}\left(W^{-1} B\right),$$

where we used the facts that W_α is positive definite and B_α is positive semi-definite. This implies that the value of the Fisher criterion increases if a data pair in the same class is made close.

Similarly, if $y_i \neq y_j$, we have $W_{i,j}^{(w)} = 0$ and $W_{i,j}^{(b)} > 0$. Then we have

$$\begin{aligned}\mathrm{tr}\left(W_\alpha^{-1}B_\alpha\right) &= \mathrm{tr}\left((W - \beta W_{i,j}^{(w)} v_{i,j}v_{i,j}^\top)^{-1}(B - \beta W_{i,j}^{(b)} v_{i,j}v_{i,j})\right)\\ &= \mathrm{tr}\left(W^{-1}B\right) - \beta W_{i,j}^{(b)}\langle W^{-1}v_{i,j}, v_{i,j}\rangle. \qquad (31)\end{aligned}$$

If $\alpha > 1$, we have $\beta < 0$ and hence Eq. (31) yields

$$\mathrm{tr}\left(W_\alpha^{-1}B_\alpha\right) > \mathrm{tr}\left(W^{-1}B\right).$$

This implies that the value of the Fisher criterion increases if a pair of samples in different classes are separated from each other.

Appendix C. Efficient Computation of T_{LFDA}

As shown in Eq. (15), the LFDA transformation matrix T_{LFDA} can be computed analytically using the generalized eigenvectors and generalized eigenvalues of the following generalized eigenvalue problem.

$$\widetilde{S}^{(b)}\widetilde{\varphi} = \widetilde{\lambda}\widetilde{S}^{(w)}\widetilde{\varphi}.$$

Given $\widetilde{S}^{(b)}$ and $\widetilde{S}^{(w)}$, the computational complexity of calculating T_{LFDA} is $O(rd^2)$. Here, we provide an efficient computing method of $\widetilde{S}^{(b)}$ and $\widetilde{S}^{(w)}$.

Let $\widetilde{S}^{(m)}$ be the *local mixture scatter matrix* defined by

$$\widetilde{S}^{(m)} \equiv \widetilde{S}^{(b)} + \widetilde{S}^{(w)}.$$

From Eqs. (9)–(11), we can immediately show that $\widetilde{S}^{(m)}$ is expressed in the following pairwise form.

$$\widetilde{S}^{(m)} = \frac{1}{2}\sum_{i,j=1}^{n} \widetilde{W}_{i,j}^{(m)}(x_i - x_j)(x_i - x_j)^\top,$$

where $\widetilde{W}^{(m)}$ is the n-dimensional matrix with (i,j)-th element being

$$\widetilde{W}_{i,j}^{(m)} \equiv \begin{cases} A_{i,j}/n & \text{if } y_i = y_j,\\ 1/n & \text{if } y_i \neq y_j.\end{cases}$$

Since

$$\begin{aligned}\widetilde{S}^{(m)} &= \frac{1}{2}\sum_{i,j=1}^{n} \widetilde{W}_{i,j}^{(m)}(x_ix_i^\top + x_jx_j^\top - x_ix_j^\top - x_jx_i^\top)\\ &= \sum_{i=1}^{n}\left(\sum_{j=1}^{n}\widetilde{W}_{i,j}^{(m)}\right)x_ix_i^\top - \sum_{i,j=1}^{n}\widetilde{W}_{i,j}^{(m)}x_ix_j^\top,\end{aligned}$$

$\widetilde{S}^{(m)}$ can be expressed in a matrix form as

$$\widetilde{S}^{(m)} = X\widetilde{L}^{(m)}X^\top, \qquad (32)$$

where

$$\widetilde{L}^{(m)} \equiv \widetilde{D}^{(m)} - \widetilde{W}^{(m)}, \tag{33}$$

and $\widetilde{D}^{(m)}$ is the n-dimensional diagonal matrix with i-th diagonal element being

$$\widetilde{D}^{(m)}_{i,i} \equiv \sum_{j=1}^{n} \widetilde{W}^{(m)}_{i,j}.$$

Similarly, $\widetilde{S}^{(w)}$ can be expressed in a matrix form as

$$\widetilde{S}^{(w)} = X\widetilde{L}^{(w)}X^{\top}, \tag{34}$$

where

$$\widetilde{L}^{(w)} \equiv \widetilde{D}^{(w)} - \widetilde{W}^{(w)}, \tag{35}$$

and $\widetilde{D}^{(w)}$ is the n-dimensional diagonal matrix with i-th diagonal element being

$$\widetilde{D}^{(w)}_{i,i} \equiv \sum_{j=1}^{n} \widetilde{W}^{(w)}_{i,j}.$$

$\widetilde{L}^{(m)}$ and $\widetilde{L}^{(w)}$ are n-dimensional matrices and could be very high dimensional. However, $\widetilde{L}^{(w)}$ can be made block-diagonal if the samples $\{x_i\}_{i=1}^{n}$ are sorted according to the labels $\{y_i\}_{i=1}^{n}$. Furthermore, diagonal sub-matrices of $\widetilde{L}^{(w)}$ can be sparse if the affinity matrix A is sparsely defined (see Appendix D for detail). Therefore, directly calculating $\widetilde{S}^{(w)}$ by Eq. (34) may be already computationally efficient.

On the other hand, computing $\widetilde{S}^{(m)}$ directly by Eq. (32) is not so efficient since $\widetilde{W}^{(m)}$ is *dense*. This problem can be alleviated as follows. $\widetilde{W}^{(m)}$ can be decomposed as

$$\widetilde{W}^{(m)} = \frac{1}{n}1_n 1_n^{\top} + \widetilde{W}^{(m_1)} + \widetilde{W}^{(m_2)},$$

where 1_n is the n-dimensional vector with all ones and $\widetilde{W}^{(m_1)}$ and $\widetilde{W}^{(m_2)}$ are the n-dimensional matrices with (i,j)-th element being

$$\widetilde{W}^{(m_1)}_{i,j} \equiv \begin{cases} A_{i,j}/n & \text{if } y_i = y_j, \\ 0 & \text{if } y_i \neq y_j, \end{cases}$$

$$\widetilde{W}^{(m_2)}_{i,j} \equiv \begin{cases} -1/n & \text{if } y_i = y_j, \\ 0 & \text{if } y_i \neq y_j. \end{cases}$$

Then $\widetilde{S}^{(m)}$ can be expressed as

$$\widetilde{S}^{(m)} = X\widetilde{D}^{(m)}X^{\top} - \frac{1}{n}X1_n(X1_n)^{\top} - X\widetilde{W}^{(m_1)}X^{\top} - X\widetilde{W}^{(m_2)}X^{\top}, \tag{36}$$

where the diagonal matrix $\widetilde{D}^{(m)}$ is expressed in terms of $\widetilde{W}^{(m_1)}$ as

$$\widetilde{D}^{(m)}_{i,i} = 1 - \frac{n_{y_i}}{n} + \sum_{j=1}^{n} \widetilde{W}^{(m_1)}_{i,j}.$$

Note that n_{y_i} in the above equation is the number of samples in the class which the sample x_i belongs to. $\widetilde{W}^{(m_2)}$ is a constant block-diagonal matrix if the samples $\{x_i\}_{i=1}^n$ are sorted according to the labels $\{y_i\}_{i=1}^n$. Therefore, $X\widetilde{W}^{(m_2)}X^\top$ in the right-hand side of Eq. (36) can be computed efficiently. Similarly, $\widetilde{W}^{(m_1)}$ can also be made block-diagonal, so $X\widetilde{W}^{(m_1)}X^\top$ in the right-hand side of Eq. (36) may also be computed efficiently; if the affinity matrix A is sparse, the computational efficiency can be further improved. The first two terms in the right-hand side of Eq. (36) can also be computed efficiently. Therefore, computing $\widetilde{S}^{(m)}$ by Eq. (36) may be more efficient than directly by Eq. (32). Finally, we can compute $\widetilde{S}^{(b)}$ efficiently by using $\widetilde{S}^{(m)}$ as

$$\widetilde{S}^{(b)} = \widetilde{S}^{(m)} - \widetilde{S}^{(w)}.$$

To further improve computational efficiency, the affinity matrix A may be computed in a class-wise manner since we do not need the affinity values for sample pairs in different classes. This speeds up the nearest neighbor search which is often carried out when defining A (see Appendix D). The nearest neighbor search itself could also be a bottleneck, but this may be eased by incorporating the prior knowledge of the data structure or by approximation (see Saul and Roweis, 2003, and references therein).

The above efficient implementation of LFDA is summarized as a pseudo code in Figure 2.

Appendix D. Definitions of Affinity Matrix

Here, we briefly review typical choices of the affinity matrix A.

D.1 Heat Kernel

A standard choice of the affinity matrix A is

$$A_{i,j} = \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma^2}\right), \tag{37}$$

where σ (> 0) is a tuning parameter which controls the 'decay' of the affinity (e.g., Belkin and Niyogi, 2003).

D.2 Euclidean Neighbor

The heat kernel gives a non-sparse affinity matrix. It would be computationally advantageous if the affinity matrix is made sparse. A sparse affinity matrix can be obtained by assigning positive affinity values only to neighboring samples. More specifically, x_i and x_j are said to be *neighbors* if

$$\|x_i - x_j\| \leq \varepsilon,$$

where ε (> 0) is a tuning parameter. Then $A_{i,j}$ is defined by Eq. (37) for two neighboring samples and $A_{i,j} = 0$ for non-neighbors (Tenenbaum et al., 2000).

This definition includes two tuning parameters (ε and σ), which are rather troublesome to determine in practice. To ease the problem, we may simply let $A_{i,j} = 1$ if x_i and x_j are neighbors and $A_{i,j} = 0$ otherwise. This corresponds to setting $\sigma = \infty$.

D.3 Nearest Neighbor

Tuning the distance threshold ε is practically rather cumbersome since the relation between the number of neighbors and the value of ε is not intuitively clear. Another option to determine the neighbors is to directly specify the number of neighbors (Roweis and Saul, 2000; Tenenbaum et al., 2000). Let $NN_i^{(K)}$ be the set of K nearest neighbor samples of x_i under the Euclidean distance, where K is a tuning parameter. If $x_j \in NN_i^{(K)}$ or $x_i \in NN_j^{(K)}$, x_i and x_j are regarded as neighbors; otherwise they are regarded as non-neighbors. Then the affinity matrix is defined by the heat kernel or in the simple zero-one manner.

D.4 Local Scaling

A drawback of the above definitions could be that the affinity is computed globally in the same way. The density of data samples may be different depending on regions. Therefore, it would be more appropriate to take the local scaling of the data into account. Following this idea, Zelnik-Manor and Perona (2005) proposed defining the affinity matrix as

$$A_{i,j} = \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma_i \sigma_j}\right).$$

σ_i represents the local scaling of the data samples around x_i, which is determined by

$$\sigma_i = \|x_i - x_i^{(K)}\|,$$

where $x_i^{(K)}$ is the K-th nearest neighbor of x_i. The parameter K is a tuning parameter, but Zelnik-Manor and Perona (2005) demonstrated that $K = 7$ works well on the whole. This would be a convenient heuristic for those who do not have any subjective/prior preferences. We employed the local scaling method with this heuristic all through the paper.

For computational efficiency, we may further sparsify the above affinity matrix based on, for example, the nearest neighbor idea, although this is not tested in this paper.

Appendix E. Pairwise Expression of $\overline{S}^{(b)}$

A pairwise expression of $\overline{S}^{(b)}$ can be derived as

$$\begin{aligned}
\overline{S}^{(b)} &= \sum_{k=1}^{n} \frac{1}{\overline{n}^{[k]}} \sum_{\ell=1}^{c} \overline{n}_\ell^{[k]} (\overline{\mu}_\ell^{[k]} - \overline{\mu}^{[k]})(\overline{\mu}_\ell^{[k]} - \overline{\mu}^{[k]})^\top \\
&= \sum_{k=1}^{n} \frac{1}{\overline{n}^{[k]}} \left(\sum_{\ell=1}^{c} \overline{n}_\ell^{[k]} \overline{\mu}_\ell^{[k]} \overline{\mu}_\ell^{[k]\top} - \overline{n}^{[k]} \overline{\mu}^{[k]} \overline{\mu}^{[k]\top} \right) \\
&= \sum_{k=1}^{n} \frac{1}{\overline{n}^{[k]}} \left(\sum_{\ell=1}^{c} \overline{n}_\ell^{[k]} \overline{\mu}_\ell^{[k]} \overline{\mu}_\ell^{[k]\top} - \sum_{i=1}^{n} \overline{A}_{i,k} \overline{x}_i \overline{x}_i^\top + \sum_{i=1}^{n} \overline{A}_{i,k} \overline{x}_i \overline{x}_i^\top - \overline{n}^{[k]} \overline{\mu}^{[k]} \overline{\mu}^{[k]\top} \right) \\
&= \frac{1}{2} \sum_{k=1}^{n} \frac{1}{\overline{n}^{[k]}} \Bigg(- \sum_{\ell=1}^{c} \frac{1}{\overline{n}_\ell^{[k]}} \sum_{i,j:y_i=y_j=\ell} \overline{A}_{i,k} \overline{A}_{j,k} (\overline{x}_i - \overline{x}_j)(\overline{x}_i - \overline{x}_j)^\top \\
&\qquad + \frac{1}{\overline{n}^{[k]}} \sum_{i,j=1}^{n} \overline{A}_{i,k} \overline{A}_{j,k} (\overline{x}_i - \overline{x}_j)(\overline{x}_i - \overline{x}_j)^\top \Bigg) \\
&= \frac{1}{2} \sum_{i,j=1}^{n} \overline{W}_{i,j}^{(b)} (\overline{x}_i - \overline{x}_j)(\overline{x}_i - \overline{x}_j)^\top,
\end{aligned}$$

which yields Eqs. (19) and (20). ■

References

A. Albert. *Regression and the Moore-Penrose Pseudoinverse*. Academic Press, New York and London, 1972.

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12(10):2385–2404, 2000.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

D. P. Bertsekas. On the Goldstein-Levitin-Polyak gradient projection method. *IEEE Transactions on Automatic Control*, 21(2):174–184, 1976.

C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, 2004.

F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, Providence, R.I., 1997.

D. Coomans, M. Broeckaert, M. Jonckheer, and D. L. Massart. Comparison of multivariate discriminant techniques for clinical data—Application to the thyroid functional state. *Methods of Information in Medicine*, 22:93–101, 1983.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, series B*, 39(1):1–38, 1977.

R. O. Duda, P. E. Hart, and D. G. Stor. *Pattern Classification*. Wiley, New York, 2001.

N. Duffy and M. Collins. Convolution kernels for natural language. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

B. S. Everitt, S. Landau, and M. Leese. *Cluster Analysis*. Arnold, London, 2001.

R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2): 179–188, 1936.

J. H. Friedman. Regularized discriminant analysis. *Journal of the American Statistical Association*, 84:165–175, 1989.

K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, Inc., Boston, second edition, 1990.

T. Gärtner. A survey of kernels for structured data. *SIGKDD Explorations*, 5(1):S268–S275, 2003.

T. Gärtner, P. Flach, and S. Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Proceedings of the Sixteenth Annual Conference on Computational Learning Theory*, 2003.

A. Globerson, G. Chechik, F. Pereira, and N. Tishby. Euclidean embedding of co-occurrence data. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 497–504. MIT Press, Cambridge, MA, 2005.

A. Globerson and S. Roweis. Metric learning by collapsing classes. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 451–458, Cambridge, MA, 2006. MIT Press.

J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 513–520. MIT Press, Cambridge, MA, 2005.

J. Ham, D. D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. In *Proceedings of the Twenty-First International Conference on Machine Learning*, New York, NY, 2004. ACM Press.

T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(6):607–615, 1996a.

T. Hastie and R. Tibshirani. Discriminant analysis by Gaussian mixtures. *Journal of the Royal Statistical Society, Series B*, 58(1):155–176, 1996b.

X. He and P. Niyogi. Locality preserving projections. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

R. E. Henkel. *Tests of Significance*. SAGE Publication, Beverly Hills, 1979.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

H. Kashima and T. Koyanagi. Kernels for semi-structured date. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 291–298, San Francisco, CA, 2002. Morgan Kaufmann.

H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *Proceedings of the Twentieth International Conference on Machine Learning*, San Francisco, CA, 2003. Morgan Kaufmann.

T. Kohonen. *Self-Organization and Associative Memory*. Springer-Verlag, Berlin, 1989.

R. I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 315–322, 2002.

S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002.

J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, Berkeley, CA., USA, 1967. University of California Press.

S. Mika, G. Rätsch, J. Weston, B. Schölkopf, A. Smola, and K.-R. Müller. Constructing descriptive and discriminative nonlinear features: Rayleigh coefficients in kernel feature spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):623–628, 2003.

G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for adaboost. *Machine Learning*, 42(3): 287–320, 2001.

S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

L. K. Saul and S. T. Roweis. Think globally, fit locally: Unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4(Jun):119–155, 2003.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society, Series B*, 36:111–147, 1974.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

G. Wahba. *Spline Model for Observational Data*. Society for Industrial and Applied Mathematics, Philadelphia and Pennsylvania, 1990.

K. Weinberger, J. Blitzer, and L. Saul. Distance metric learning for large margin nearest neighbor classification. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1473–1480, Cambridge, MA, 2006. MIT Press.

L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1601–1608. MIT Press, Cambridge, MA, 2005.

Journal of Machine Learning Research 8 (2007) 1063-1095 Submitted 10/06; Published 5/07

Undercomplete Blind Subspace Deconvolution

Zoltán Szabó SZZOLI@CS.ELTE.HU
Barnabás Póczos PBARN@CS.ELTE.HU
András Lőrincz ANDRAS.LORINCZ@ELTE.HU
Department of Information Systems
Eötvös Loránd University
Pázmány P. sétány 1/C
Budapest H-1117, Hungary

Editor: Michael Jordan

Abstract

We introduce the blind subspace deconvolution (BSSD) problem, which is the extension of both the blind source deconvolution (BSD) and the independent subspace analysis (ISA) tasks. We examine the case of the undercomplete BSSD (uBSSD). Applying temporal concatenation we reduce this problem to ISA. The associated 'high dimensional' ISA problem can be handled by a recent technique called joint f-decorrelation (JFD). Similar decorrelation methods have been used previously for kernel independent component analysis (kernel-ICA). More precisely, the kernel canonical correlation (KCCA) technique is a member of this family, and, as is shown in this paper, the kernel generalized variance (KGV) method can also be seen as a decorrelation method in the feature space. These kernel based algorithms will be adapted to the ISA task. In the numerical examples, we (i) examine how efficiently the emerging higher dimensional ISA tasks can be tackled, and (ii) explore the working and advantages of the derived kernel-ISA methods.

Keywords: undercomplete blind subspace deconvolution, independent subspace analysis, joint decorrelation, kernel methods

1. Introduction

Independent component analysis (ICA) (Jutten and Herault, 1991; Comon, 1994) aims to recover linearly or non-linearly mixed independent and hidden sources. There is a broad range of applications for ICA, such as blind source separation, feature extraction and denoising. Particular applications include the analysis of financial and neurobiological data, fMRI, EEG, and MEG. For recent review concerning ICA the reader is referred to the literature (Hyvärinen et al., 2001; Cichocki and Amari, 2002).

Traditional ICA algorithms are one-dimensional in the sense that all sources are assumed to be independent real valued random variables. Nonetheless, applications in which only certain groups of sources are independent may be highly relevant in practice. In this case, the independent sources can be multidimensional. For instance, consider the generalization of the cocktail-party problem, where *independent groups of people* are talking about independent topics or more than one *group of musicians* is playing at the party. The separation task requires an extension of ICA, which can be called independent subspace analysis (ISA) (Hyvärinen and Hoyer, 2000), multidimensional independent component analysis (MICA) (Cardoso, 1998), and group ICA (Theis, 2005a). We will use the first of these abbreviations throughout this paper.

Strenuous efforts have been made to develop ISA algorithms (Cardoso, 1998; Akaho et al., 1999; Hyvärinen and Hoyer, 2000; Hyvärinen and Köster, 2006; Vollgraf and Obermayer, 2001; Bach and Jordan, 2003; Stögbauer et al., 2004; Póczos and Lőrincz, 2005b,a; Theis, 2005a,b, 2006; Szabó and Lőrincz, 2006; Nolte et al., 2006). For the most part, ISA-related theoretical problems concern the estimation of entropy or of mutual information. For this, the k-nearest neighbors (Póczos and Lőrincz, 2005b) and the geodesic spanning tree methods (Póczos and Lőrincz, 2005a) can be applied. Other recent approaches seek independent subspaces via kernel methods (Bach and Jordan, 2003) and joint block diagonalization (Theis, 2005a, 2006).

Another extension of the original ICA task is the blind source deconvolution (BSD) problem. Such a problem emerges, for example, at a cocktail-party being held in an *echoic room*. Several BSD algorithms were developed in the past. See, for example, the review of Pedersen et al. (2007). Like ICA, BSD has several applications: (i) remote sensing applications; passive radar/sonar processing (MacDonald and Cain, 2005; Hedgepeth et al., 1999), (ii) image-deblurring, image restoration (Vural and Sethares, 2006), (iii) speech enhancement using microphone arrays, acoustics (Douglas et al., 2005; Mitianoudis and Davies, 2003; Roan et al., 2003; Araki et al., 2003), (iv) multi-antenna wireless communications, sensor networks (Akyildiz et al., 2002; Deligianni et al., 2006), (v) biomedical signal—EEG, ECG, MEG, fMRI—analysis (Jung et al., 2000; Glover, 1999; Dyrholm et al., 2007), (vi) optics (Kotzer et al., 1998), (vii) seismic exploration (Karslı, 2006).

The simultaneous assumption of the two extensions, that is, ISA combined with BSD, seems to be a more realistic model than either of the two models alone. For example, at the cocktail-party, groups of people or groups of musicians may form independent source groups and echoes could be present. This task will be called blind subspace deconvolution (BSSD). We treat the undercomplete case (uBSSD) here. In terms of the cocktail-party problem, it is assumed that there are more microphones than acoustic sources. Here we note that the complete, and in particular the overcomplete, BSSD task is challenging and as of yet no general solution is known. We can show that temporal concatenation turns the uBSSD task into an ISA problem. One of the most stringent applications of BSSD could be the analysis of EEG or fMRI signals. The ICA assumptions could be highly problematic here, because some sources may depend one another, so an ISA model seems better. Furthermore, the passing of information from one area to another and the related delayed and transformed activities may be modeled as echoes. Thus, one can argue that BSSD may fit this important problem domain better than ICA or even ISA.

In principle, the ISA problem can be treated with the methods listed above. However, the dimension of the ISA problem derived from an uBSSD task is not amenable to state-of-the-art ISA methods. According to a recent decomposition principle, the ISA Separation Theorem (Szabó et al., 2006b), the ISA task can be divided into two consecutive steps under certain conditions: after the application of the ICA algorithm, the ICA elements need to be grouped.[1] The importance of this direction stems from the fact that ICA methods can deal with problems in high dimensions. The derived ISA task will be solved with the use of the decomposition principle augmented by the joint f-decorrelation (JFD) technique (Szabó and Lőrincz, 2006).

We show other ISA approaches beyond the JFD method: We adapted the kernel canonical correlation analysis (KCCA) and the kernel generalized variance (KGV) methods (Bach and Jordan, 2002) to measure the mutual dependency of multidimensional variables. One can show that simi-

1. The possibility of such a decomposition principle was suspected by Cardoso (1998), who based his conjecture on numerical experiments. To the best of our knowledge, a proof encompassing sufficient conditions for this intriguing hypothesis was first published by Szabó et al. (2006a).

larly to the JFD and the KCCA methods, the KGV technique deals with nonlinear decorrelation in function spaces. We found that they can be more precise but are limited to smaller problems.

The paper is structured as follows: Section 2 formulates the problem domain. Section 3 shows how to transform the uBSSD task into an ISA task. The JFD method, which we use to solve the derived ISA task, is the subject of Section 4. This section also addresses how to tailor the KCCA and KGV kernel-ICA methods to solve the ISA problem. Section 5 contains the numerical illustrations and conclusions are drawn in Section 6.

2. The BSSD and the ISA Model

The BSSD task and its special case, the ISA model, are defined in Section 2.1. Section 2.2 details the ambiguities of the ISA task. Section 2.3 introduces some possible ISA cost functions.

2.1 The BSSD Equations

Here, we define the BSSD task. Assume that we have M hidden, independent, multidimensional *components* (random variables). Suppose also that only their casual FIR filtered mixture is available for observation:[2]

$$\mathbf{x}(t) = \sum_{l=0}^{L} \mathbf{H}_l \mathbf{s}(t-l), \tag{1}$$

where $\mathbf{s}(t) = \left[\mathbf{s}^1(t);\ldots;\mathbf{s}^M(t)\right] \in \mathbb{R}^{Md}$ is a vector concatenated of components $\mathbf{s}^m(t) \in \mathbb{R}^d$. For a given m, $\mathbf{s}^m(t)$ is i.i.d. (independent and identically distributed) in time t, $\mathbf{s}^m$s are non-Gaussian, and $I(\mathbf{s}^1,\ldots,\mathbf{s}^M) = 0$, where I stands for the mutual information of the arguments. The total dimension of the components is $D_s := Md$, the dimension of the observation $\mathbf{x}$ is D_x. Matrices $\mathbf{H}_l \in \mathbb{R}^{D_x \times D_s}$ $(l = 0,\ldots,L)$ describe the mixing, these are the *mixing matrices*. Without any loss of generality it may be assumed that $E[\mathbf{s}] = \mathbf{0}$, where E denotes the expectation value. Then $E[\mathbf{x}] = \mathbf{0}$ holds, as well. The goal of the BSSD problem is to estimate the original source $\mathbf{s}(t)$ by using observations $\mathbf{x}(t)$ only.

The case $L = 0$ corresponds to the ISA task, and if $d = 1$ also holds then the ICA task is recovered. In the BSD task $d = 1$ and L is a non-negative integer. $D_x > D_s$ is the *undercomplete*, $D_x = D_s$ is the *complete*, and $D_x < D_s$ is the *overcomplete* task. Here, we treat the undercomplete BSSD (uBSSD) problem. We will transform the uBSSD task to undercomplete ISA (uISA) or to complete ISA. From now on they both will be called ISA.

Note 1 *Mixing matrices* $\mathbf{H}_l$ $(0 \le l \le L)$ *have a one-to-one mapping to polynomial matrix*[3] $\mathbf{H}[z] := \sum_{l=0}^{L} \mathbf{H}_l z^{-l} \in \mathbb{R}[z]^{D_x \times D_s}$, *where* z *is the time-shift operation, that is* $(z^{-1}\mathbf{u})(t) = \mathbf{u}(t-1)$. $\mathbf{H}[z]$ *may be regarded as an operation that maps* D_s*-dimensional series to* D_x*-dimensional series. Equation* (1) *can be written as* $\mathbf{x} = \mathbf{H}[z]\mathbf{s}$.

Note 2 *It can be shown (Rajagopal and Potter, 2003) that in the uBSSD task* $\mathbf{H}[z]$ *has a polynomial matrix left inverse* $\mathbf{W}[z] \in \mathbb{R}[z]^{D_s \times D_x}$ *with probability 1, under mild conditions. In other words, for these polynomial matrices* $\mathbf{W}[z]$ *and* $\mathbf{H}[z]$, $\mathbf{W}[z]\mathbf{H}[z]$ *is the identity mapping. The mild condition is as follows: Coefficients of polynomial matrix* $\mathbf{H}[z]$, *that is, random matrix* $[\mathbf{H}_0;\ldots;\mathbf{H}_L]$ *is drawn from*

2. Causal: $l \ge 0$ in $\sum_l$. FIR: the number of terms in the sum is finite.
3. $\mathbf{H}[z]$ is also known as *channel matrix* or *transfer function* in the literature.

a continuous distribution. Under this condition, hidden source $\mathbf{s}(t)$ *can be estimated by a* suitable *causal FIR filtered form of observation* $\mathbf{x}(t)$.

For the uBSSD task it is assumed that $\mathbf{H}[z]$ has a polynomial matrix left inverse. For the uISA and ISA tasks it is supposed that *mixing matrix* $\mathbf{H}_0 \in \mathbb{R}^{D_x \times D_s}$ has full column rank, that is its rank is D_s.

2.2 Ambiguities of the ISA Model

Because the uBSSD task will be reduced to ISA, it is important to see the ambiguities of the ISA task. First, the complete ISA problem ($L = 0, D_x = D_s$) is presented, the undercomplete ISA will be treated later.

The identification of the ISA model is ambiguous. However, the ambiguities are simple (Theis, 2004): hidden multidimensional components can be determined up to permutation and up to invertible transformation within the subspaces. Ambiguities within the subspaces can be weakened. Namely, because of the invertibility of mixing matrix $\mathbf{H}[z] = \mathbf{H}_0 \in \mathbb{R}^{D_s \times D_s}$, it can be assumed without any loss of generality that both the sources and the observation are *white*, that is,

$$E[\mathbf{s}] = \mathbf{0}, cov\,[\mathbf{s}] = \mathbf{I}_{D_s},$$
$$E[\mathbf{x}] = \mathbf{0}, cov\,[\mathbf{x}] = \mathbf{I}_{D_x},$$

where $\mathbf{I}_{D_s}$ is the D_s-dimensional identity matrix and cov is the covariance matrix. It then follows that the mixing matrix $\mathbf{H}_0$ and thus the *demixing matrix* $\mathbf{W} = \mathbf{H}_0^{-1}$ are orthogonal:

$$\mathbf{I}_{D_s} = cov[\mathbf{x}] = E\left[\mathbf{x}\mathbf{x}^*\right] = \mathbf{H}_0 E\left[\mathbf{s}\mathbf{s}^*\right] \mathbf{H}_0^* = \mathbf{H}_0 \mathbf{I}_{D_s} \mathbf{H}_0^* = \mathbf{H}_0 \mathbf{H}_0^*,$$

where $*$ denotes transposition. In sum, $\mathbf{H}_0, \mathbf{W} \in \mathcal{O}^{D_s}$, where $\mathcal{O}^{D_s}$ denotes the set of D_s-dimensional orthogonal matrices. Now, $\mathbf{s}^m$ sources are determined up to permutation and orthogonal transformation.

In order to transform the undercomplete ISA task into a complete ISA task with white observations let $\mathbf{C} := cov[\mathbf{x}] = E[\mathbf{x}\mathbf{x}^*] = \mathbf{H}_0\mathbf{H}_0^* \in \mathbb{R}^{D_x \times D_x}$ denote the covariance matrix of the observation. Rank of $\mathbf{C}$ is D_s, since the rank of matrix $\mathbf{H}_0$ is D_s according to our assumptions. Matrix $\mathbf{C}$ is symmetric ($\mathbf{C} = \mathbf{C}^*$), thus it can be decomposed as follows: $\mathbf{C} = \mathbf{U}\mathbf{D}\mathbf{U}^*$, where $\mathbf{U} \in \mathbb{R}^{D_x \times D_s}$, and the columns of matrix $\mathbf{U}$ are orthogonal, that is, $\mathbf{U}^*\mathbf{U} = \mathbf{I}_{D_s}$. Furthermore, the rank of diagonal matrix $\mathbf{D} \in \mathbb{R}^{D_s \times D_s}$ is D_s. The principal component analysis can provide a decomposition in the desired form. Let $\mathbf{Q} := \mathbf{D}^{-1/2}\mathbf{U}^* \in \mathbb{R}^{D_s \times D_x}$. Then the original observation $\mathbf{x}$ can be modified to $\mathbf{x}' := \mathbf{Q}\mathbf{x} = \mathbf{Q}\mathbf{H}_0\mathbf{s} \in \mathbb{R}^{D_s}$. The resulting $\mathbf{x}'$ is white and can be regarded as the observation of a *complete* ISA task having mixing matrix $\mathbf{Q}\mathbf{H}_0 \in \mathcal{O}^{D_s}$.

2.3 ISA Cost Functions

After the whitening procedure (Section 2.2), the ISA task can be viewed as the minimization of the mutual information between the estimated components on the orthogonal group:

$$J_I(\mathbf{W}) := I\left(\mathbf{y}^1, \ldots, \mathbf{y}^M\right), \tag{2}$$

where $\mathbf{y} = \mathbf{W}\mathbf{x}$, $\mathbf{y} = \left[\mathbf{y}^1; \ldots; \mathbf{y}^M\right]$, $\mathbf{y}^m \in \mathbb{R}^d$, and $\mathbf{W} \in O^D$. This formulation of the ISA task serves us in Section 4.1, where we estimate the dependencies of the multidimensional variables.

The ISA task can be rewritten into the minimization of the sum of Shannon's multidimensional differential entropies (Póczos and Lőrincz, 2005b):

$$J_H(\mathbf{W}) := \sum_{m=1}^{M} H(\mathbf{y}^m), \tag{3}$$

where $\mathbf{y} = \mathbf{W}\mathbf{x}$, $\mathbf{y} = \left[\mathbf{y}^1; \ldots; \mathbf{y}^M\right]$, $\mathbf{y}^m \in \mathbb{R}^d$, $\mathbf{W} \in \mathcal{O}^D$.

Note 3 *Until now, we formulated the ISA task by means of the entropy or the mutual information of multidimensional random variables, see Equations* (2) *and* (3). *However, any algorithm that treats mutual information between* 1-dimensional *random variables can also be sufficient. This statement is based on the considerations below. Well-known identities of mutual information and entropy expressions (Cover and Thomas, 1991) show that the minimization of cost function*

$$J_{H,I}(\mathbf{W}) := \sum_{m=1}^{M}\sum_{i=1}^{d} H(y_i^m) - \sum_{m=1}^{M} I(y_1^m, \ldots, y_d^m),$$

or that of

$$J_{I,I}(\mathbf{W}) := I\left(y_1^1, \ldots, y_d^M\right) - \sum_{m=1}^{M} I\left(y_1^m, \ldots, y_d^m\right)$$

can also solve the ISA task. Here, $\mathbf{y} = \mathbf{W}\mathbf{x}$ *is the estimated ISA source, where* $\mathbf{x} \in \mathbb{R}^D$ *is the whitened observation in the ISA model.* $\mathbf{W} \in \mathcal{O}^D$ *is the estimated ISA demixing matrix, and in* $\mathbf{y} = \left[\mathbf{y}^1; \ldots; \mathbf{y}^M\right] \in \mathbb{R}^D$ *the* $\mathbf{y}^m \in \mathbb{R}^d$*,* $m = 1, \ldots, M$*, represent the estimated components with coordinates* $y_i^m \in \mathbb{R}$*. The first term of both cost functions* $J_{H,I}$ *and* $J_{I,I}$ *is an ICA cost function. Thus, these first terms can be fixed by means of ICA preprocessing.*[4] *In this case, if the Separation Theorem holds (for details see Section 3.2), then term* $\sum_{m=1}^{M} I(y_1^m, \ldots, y_d^m)$ *implies that the maximization of the sum of mutual information between* 1-dimensional *random variables* within *the subspaces is sufficient for solving the ISA task.*

3. Reduction Steps

Here we show that the direct search for inverse FIR filter can be circumvented (Note 2). Namely, temporal concatenation reduces the uBSSD task to an (u)ISA problem (Section 3.1). Our earlier results will allow further simplifications. We will reduce the ISA task to an ICA task plus a search for optimal permutation of the ICA coordinates. This decomposition principle will be elaborated in Section 3.2 by means of the Separation Theorem.

3.1 Reduction of uBSSD to (u)ISA

We reduce the uBSSD task to an ISA problem. The BSD literature provides the basis for our reduction; Févotte and Doncarli (2003) use temporal concatenation in their work. This method can be extended to multidimensional $\mathbf{s}^m$ components in a natural fashion:

Let L' be such that

$$D_x L' \geq D_s(L + L') \tag{4}$$

4. From the algorithmic point of view, *any* ICA algorithm that minimizes cost function $I(y_1^1, \ldots, y_d^M)$ suits the ICA preprocessing step.

is fulfilled. Such L' exists due to the undercomplete assumption $D_x > D_s$:

$$L' \geq \left\lceil \frac{D_s L}{D_x - D_s} \right\rceil. \tag{5}$$

This choice of L' guarantees that the reduction gives rise to an (under)complete ISA task: let $x_m(t)$ denote the m^{th} coordinate of observation $\mathbf{x}(t)$ and let the matrix $\mathbf{H}_l \in \mathbb{R}^{D_x \times Md}$ be decomposed into $1 \times d$ sized blocks. That is, $\mathbf{H}_l = [\mathbf{H}_l^{ij}]_{i=1..D_x, j=1..M}$ ($\mathbf{H}_l^{ij} \in \mathbb{R}^{1\times d}$), where i and j denote row and column indices, respectively. Using notations

$$\begin{aligned}
\mathbf{S}^m(t) &:= [\mathbf{s}^m(t); \mathbf{s}^m(t-1); \ldots; \mathbf{s}^m(t-(L+L')+1)] \in \mathbb{R}^{d(L+L')},\\
\mathbf{X}^m(t) &:= [x_m(t); x_m(t-1); \ldots; x_m(t-L'+1)] \in \mathbb{R}^{L'},\\
\mathbf{S}(t) &:= [\mathbf{S}^1(t); \ldots; \mathbf{S}^M(t)] \in \mathbb{R}^{Md(L+L')=D_s(L+L')},\\
\mathbf{X}(t) &:= [\mathbf{X}^1(t); \ldots; \mathbf{X}^{D_x}(t)] \in \mathbb{R}^{D_x L'},\\
\mathbf{A}^{ij} &:= \begin{bmatrix} \mathbf{H}_0^{ij} & \ldots & \mathbf{H}_L^{ij} & \mathbf{0} & \ldots & \mathbf{0} \\ & \ddots & & \ddots & & \\ & & \ddots & & \ddots & \\ \mathbf{0} & \ldots & \mathbf{0} & \mathbf{H}_0^{ij} & \ldots & \mathbf{H}_L^{ij} \end{bmatrix} \in \mathbb{R}^{L' \times d(L+L')},\\
\mathbf{A} &:= [\mathbf{A}^{ij}]_{i=1..D_x, j=1..M} \in \mathbb{R}^{D_x L' \times Md(L+L') = D_x L' \times D_s(L+L')},
\end{aligned}$$

model

$$\mathbf{X}(t) = \mathbf{A}\mathbf{S}(t) \tag{6}$$

can be obtained. Here, $\mathbf{s}^m(t)$s are i.i.d. in time t, they are independent for different m values, and Equation (4) holds for L'. Thus, (6) is either an undercomplete or a complete ISA task, depending on the relation of the l.h.s and the r.h.s of (4): the task is complete if the two sides are equal. The number of the components and the dimension of the components in task (6) are $M(L+L')$ and d, respectively.

If we end up with an undercomplete ISA problem in (6) then it can be reduced to a complete one, as was shown in Section 2.2. Thus, choosing the minimal value for L' in (5), the dimension of the obtained ISA task is

$$D_{\mathrm{ISA}} := D_s(L+L') = D_s\left(L + \left\lceil \frac{D_s L}{D_x - D_s} \right\rceil\right). \tag{7}$$

Taking into account the ambiguities of the ISA task (Section 2.2), the original $\mathbf{s}^m$ components will occur $L+L^{'}$ times and up to orthogonal transformations. As a result, in the ideal case, our estimations are as follows

$$\hat{\mathbf{s}}_k^m := \mathbf{F}_k^m \mathbf{s}^m \in \mathbb{R}^d,$$

where $k = 1, \ldots, L+L'$, $\mathbf{F}_k^m \in \mathcal{O}^d$.

3.2 Reduction of ISA to ICA

The Separation Theorem (Szabó et al., 2006b) conjectured by Cardoso (1998) allows one to decompose the solution of the ISA problem, under certain conditions, into 2 steps: In the first step, ICA estimation is executed by minimizing $I(y_1^1,\dots,y_d^M)$. In the second step, the ICA elements are grouped by finding an optimal permutation. This principle will be formalized in Section 3.2.1. Section 3.2.2 provides sufficient conditions for the theorem.

3.2.1 THE ISA SEPARATION THEOREM

We state the ISA Separation Theorem for components having possibly different d_m dimensions:

Theorem 1 (Separation Theorem for ISA) *Let $\mathbf{y}=[y_1;\dots;y_D]=\mathbf{W}\mathbf{x}\in\mathbb{R}^D$, where $\mathbf{W}\in O^D$, $\mathbf{x}\in\mathbb{R}^D$ is the whitened observation of the ISA model, and $D=\sum_{m=1}^M d_m$. Let $\mathcal{S}^{d_m}$ denote the surface of the d_m-dimensional unit sphere, that is $\mathcal{S}^{d_m}:=\{\mathbf{w}\in\mathbb{R}^{d_m}:\sum_{i=1}^{d_m}w_i^2=1\}$.*

Presume that the $\mathbf{u}:=\mathbf{s}^m\in\mathbb{R}^{d_m}$ sources $(m=1,\dots,M)$ of the ISA model satisfy condition

$$H\left(\sum_{i=1}^{d_m}w_iu_i\right)\geq\sum_{i=1}^{d_m}w_i^2H\left(u_i\right),\forall\mathbf{w}\in\mathcal{S}^{d_m}, \tag{8}$$

and that the ICA cost function $J_{ICA}(\mathbf{W})=\sum_{i=1}^D H(y_i)$ has minimum over the orthogonal matrices in $\mathbf{W}_{\text{ICA}}$. Then it is sufficient to search for the solution to the ISA task as a permutation of the solution of the ICA task. Using the concept of demixing matrices, it is sufficient to explore forms

$$\mathbf{W}_{\text{ISA}}=\mathbf{P}\mathbf{W}_{\text{ICA}},$$

where $\mathbf{P}\in\mathbb{R}^{D\times D}$ is a permutation matrix to be determined and $\mathbf{W}_{\text{ISA}}$ is the ISA demixing matrix.

The proof of the theorem is presented in Appendix A. It is intriguing that if (8) is satisfied then the simple decomposition principle provides the *global* minimum of (2). In the literature on joint block diagonalization (JBD) Abed-Meraim and Belouchrani (2004) have put forth a similar *conjecture* recently. According to this conjecture, for quadratic cost function, if Jacobi optimization is applied, the block-diagonalization of the matrices can be found by the optimization of permutations following the joint diagonalization of the matrices. ISA solutions formulated within the JBD framework (Theis, 2005a,b, 2006; Szabó and Lőrincz, 2006) make efficient use of this idea in practice. Theis (2006) could justify this approach for *local* minimum points.

3.2.2 SUFFICIENT CONDITIONS OF THE ISA SEPARATION THEOREM

The question of which types of sources satisfy the Separation Theorem is open. Equation (8) provides only a sufficient condition. Below, we list sources $\mathbf{s}^m$ that satisfy (8). Details and the extension of the Separation Theorem for complex variables can be found in a technical report of Szabó et al. (2006b).

1. Assume that variables $\mathbf{u}=\mathbf{s}^m$ satisfy the so-called w-EPI condition (EPI is shorthand for the *entropy power inequality*, Cover and Thomas, 1991), that is,

$$e^{2H\left(\sum_{i=1}^d w_iu_i\right)}\geq\sum_{i=1}^d e^{2H(w_iu_i)},\forall\mathbf{w}\in\mathcal{S}^d. \tag{9}$$

   Then inequality (8) holds for these variables too. The proof can be found in Lemma 1 of Appendix A.

2. The (9) w-EPI condition is valid

   (a) for spherically symmetric or shortly spherical variables (Fang et al., 1990). The distribution of such variables is invariant for orthogonal transformations.[5] Sketch of the proof ($\mathbf{u} = \mathbf{s}^m$): the w-EPI condition concerns projections to unit vectors. For spherical variables, the distribution and thus the entropy of these projections are independent of $\mathbf{w} \in \mathcal{S}^d$. Because $e^{2H(w_i u_i)} = e^{2H(u_i)} w_i^2$ and $\mathbf{w} \in \mathcal{S}^d$, the w-EPI is satisfied with equality $\forall \mathbf{w} \in \mathcal{S}^d$. $\square$

   (b) for 2-dimensional variables invariant to 90° rotation. Under this condition, density function h of component $\mathbf{s}^m$ is subject to the following invariance

$$h(u_1,u_2) = h(-u_2,u_1) = h(-u_1,-u_2) = h(u_2,-u_1) \quad \left(\forall \mathbf{u} \in \mathbb{R}^2\right).$$

   Sketch of the proof ($\mathbf{u} = \mathbf{s}^m$): Assume that function $f : \mathcal{S}^2 \ni \mathbf{w} \mapsto H\left(\sum_{i=1}^d w_i u_i\right)$ has global minimum on set $\mathcal{S}^2 \cap \{\mathbf{w} \geq \mathbf{0}\}$.[6] Let this minimum be at $\mathbf{w}_m \in \mathbb{R}^2$. Then, the 90° invariance warrants that function f take its global minimum also on $\mathbf{w}_m^\perp \in \mathbb{R}^2$, which is perpendicular to $\mathbf{w}_m$. Let $(\mathbf{C}^m)^* = [\mathbf{w}_m, \mathbf{w}_m^\perp] \in \mathcal{O}^2$. Now, we can estimate variables $\mathbf{C}^m \mathbf{s}^m$. This is sufficient because the ISA solution is ambiguous up to orthogonal transformations within each subspace. $\square$

   A special case of this requirement is invariance to permutation and sign changes

$$h(\pm u_1, \pm u_2) = h(\pm u_2, \pm u_1).$$

   In other words, there exists a function $g : \mathbb{R}^2 \to \mathbb{R}$, which is symmetric in its variables and

$$h(\mathbf{u}) = g(|u_1|, |u_2|).$$

   Special cases within this family are distributions

$$h(\mathbf{u}) = g\left(\sum_i |u_i|^p\right) \quad (p > 0),$$

   which are constant over the spheres of L^p-space. They are called L^p spherical variables which, for $p = 2$, corresponds to spherical variables.

   (c) for certain weakly dependent variables: Takano (1995) has determined sufficient conditions when EPI holds.[7] If the EPI property is satisfied on unit sphere $\mathcal{S}^d$, then the ISA Separation Theorem holds (Lemma 1).

These results are summarized schematically in Table 1.

5. In the ISA task the non-degenerate affine transformations of spherical variables, the so called elliptical variables, do not provide valuable generalizations due to the ambiguities of the ISA task.
6. Relation $\mathbf{w} \geq \mathbf{0}$ concerns each coordinate.
7. The constraint of $d = 2$ may be generalized to higher dimensions. We are not aware of such generalizations.

$$\begin{tikzcd}
 & \text{invariance to } 90^\circ \text{ rotation } (d=2) \arrow[ddd, Rightarrow] \arrow[dr, "\text{specially}"] & \\
 & & \text{invariance to sign and permutation} \arrow[d, "\text{specially}"] \\
 & & L^p \text{ spherical } (p>0) \\
\begin{array}{c}\text{Takano's dependency}\\ (d=2)\end{array} \arrow[r, Rightarrow] & \text{w-EPI} \arrow[d, Rightarrow] & \text{spherical symmetry} \arrow[l, Rightarrow] \arrow[u, "\text{generalization for } d=2"'] \\
 & \begin{array}{c}\text{Equation (8): sufficient}\\ \text{for the ISA Separation Theorem}\end{array} &
\end{tikzcd}$$

Table 1: Sufficient conditions for the ISA Separation Theorem.

4. ISA Methods

We showed how to convert the uBSSD task to an ISA task in Section 3.1. In the following we will present methods that can solve the ISA task. In Section 4.1 we treat estimations of the mutual information of the ISA cost functions in Section 2.3. Methods that can optimize these cost functions are elaborated in Section 4.2. We also present here the pseudocode of the procedures studied. In Section 4.3 we review methods that can treat non-equal or unknown component dimensions. In what follows, and in accordance with (1), let $\mathbf{x} \in \mathbb{R}^D$ denote the whitened observation, while $\mathbf{y} = [\mathbf{y}^1; \ldots; \mathbf{y}^M] = \mathbf{W}\mathbf{x} \in \mathbb{R}^D$ ($\mathbf{W} \in \mathcal{O}^D$) and $\mathbf{y}^m \in \mathbb{R}^d$ $(m = 1, \ldots, M)$ stand for the estimated source and its components in the ISA task, respectively.

4.1 Dependency Estimations

Here we introduce two dependency estimators. First, in Section 4.1.1 we describe a decorrelation method that uses a set of functions jointly. This method is called joint f-decorrelation (JFD) method (Szabó and Lőrincz, 2006). Our second technique (Section 4.1.2) generalizes earlier kernel-ICA methods for the ISA task. The motivation for this latter method is the efficiency and precision of kernel-ICA methods in finding independent components (Bach and Jordan, 2002). Our experiences are similar with kernel-ISA methods, see Section 5.3. We found that kernel-ISA methods need more computations, but can provide more precise solutions than the JFD technique.

4.1.1 THE JFD METHOD

The JFD method estimates the hidden $\mathbf{s}^m$ components through the decorrelation over a function set $\mathcal{F}(\ni \mathbf{f})$ (Szabó and Lőrincz, 2006). Formally, let the empirical $\mathbf{f}$-covariance matrix of $\mathbf{y}(t)$ and $\mathbf{y}^m(t)$

for function $\mathbf{f} = [\mathbf{f}^1;\ldots;\mathbf{f}^M] \in \mathcal{F}$ over $t = 1,\ldots,T$ be denoted by

$$\begin{aligned}
\Sigma(\mathbf{f},T,\mathbf{W}) &= \widehat{cov}\left[\mathbf{f}(\mathbf{y}),\mathbf{f}(\mathbf{y})\right] = \\
&= \frac{1}{T}\sum_{t=1}^{T}\left\{\mathbf{f}[\mathbf{y}(t)] - \frac{1}{T}\sum_{k=1}^{T}\mathbf{f}[\mathbf{y}(k)]\right\}\left\{\mathbf{f}[\mathbf{y}(t)] - \frac{1}{T}\sum_{k=1}^{T}\mathbf{f}[\mathbf{y}(k)]\right\}^{*}, \\
\Sigma^{i,j}(\mathbf{f},T,\mathbf{W}) &= \widehat{cov}\left[\mathbf{f}^i\left(\mathbf{y}^i\right),\mathbf{f}^j\left(\mathbf{y}^j\right)\right] = \\
&= \frac{1}{T}\sum_{t=1}^{T}\left\{\mathbf{f}^i[\mathbf{y}^i(t)] - \frac{1}{T}\sum_{k=1}^{T}\mathbf{f}^i[\mathbf{y}^i(k)]\right\}\left\{\mathbf{f}^j[\mathbf{y}^j(t)] - \frac{1}{T}\sum_{k=1}^{T}\mathbf{f}^j[\mathbf{y}^j(k)]\right\}^{*}.
\end{aligned}$$

Then, the joint decorrelation on $\mathcal{F}$ can be formulated as the minimization of cost function

$$J_{\mathrm{JFD}}(\mathbf{W}) := \sum_{\mathbf{f}\in\mathcal{F}} \|\mathbf{N}\circ\Sigma(\mathbf{f},T,\mathbf{W})\|_F^2. \tag{10}$$

Here: (i) $\mathbf{W} \in \mathcal{O}^D$, (ii) $\mathcal{F}$ denotes a set of $\mathbb{R}^D \to \mathbb{R}^D$ functions, and each function acts on each coordinate separately, (iii) $\circ$ denotes the point-wise multiplication, called the Hadamard-product, (iv) $\mathbf{N}$ masks according to the subspaces, $\mathbf{N} := \mathbf{E}_D - \mathbf{I}_M \otimes \mathbf{E}_d$, where all elements of matrix $\mathbf{E}_D \in \mathbb{R}^{D\times D}$ and $\mathbf{E}_d \in \mathbb{R}^{d\times d}$ are equal to 1, $\otimes$ is the Kronecker-product, (v) $\|\cdot\|_F^2$ denotes the square of the Frobenius norm, that is, the sum of the squares of the elements.

Cost function (10) can be interpreted as follows: for *any* function $\mathbf{f}^m : \mathbb{R}^d \to \mathbb{R}^d$ that acts on independent variables $\mathbf{y}^m$ $(m = 1,\ldots,M)$ the variables $\mathbf{f}^m(\mathbf{y}^m)$ remain independent. Thus, covariance matrix $\Sigma(\mathbf{f},T,\mathbf{W})$ of variable $\mathbf{f}(\mathbf{y}) = \left[\mathbf{f}^1\left(\mathbf{y}^1\right);\ldots;\mathbf{f}^M\left(\mathbf{y}^M\right)\right]$ is block-diagonal. Independence of estimated sources $\mathbf{y}^m$ is gauged by the uncorrelatedness on the function set $\mathcal{F}$. Thus, the non-block-diagonal portions $(\Sigma^{i,j}(\mathbf{f},T,\mathbf{W}), i \neq j)$ of covariance matrices $\Sigma(\mathbf{f},T,\mathbf{W})$ are punished. This principle is expressed by the term $\|\mathbf{N}\circ\Sigma(\mathbf{f},T,\mathbf{W})\|_F^2$.

4.1.2 KERNEL-ISA METHODS

Two alternatives for the ISA cost function of (10) are presented. They estimate the mutual information based ISA cost defined in (2) via kernels: the KCCA and KGV kernel-ICA methods of Bach and Jordan (2002) are extended to the ISA task. The original methods estimate pair-wise independence between *1-dimensional* random variables.[8] The extension to the multidimensional case is straightforward, the arguments of the kernels can be modified to multidimensional variables and the derivation of Bach and Jordan (2002) can be followed. The main steps are provided below for the sake of completeness. The resulting expressions can be used for the estimation of dependence between multidimensional random variables. The performance of these simple extensions on the related ISA applications is shown in Section 5.3.2.

The KCCA Method First, the 2-variable-case is treated and then it will be generalized to many variables.

2-variable-case Assume that the mutual dependence of two random variables $\mathbf{u} \in \mathbb{R}^{d_1}$ and $\mathbf{v} \in \mathbb{R}^{d_2}$ has to be measured. Let positive semi-definite kernels $k^{\mathbf{u}}(\cdot,\cdot) : \mathbb{R}^{d_1} \times \mathbb{R}^{d_1} \to \mathbb{R}$, and $k^{\mathbf{v}}(\cdot,\cdot) : \mathbb{R}^{d_2} \times$

8. We note that if our observations are generated by an ISA model then—unlike in the ICA task when $d = 1$—pairwise independence is *not* equivalent to mutual independence (Comon, 1994; Póczos and Lőrincz, 2005a). Nonetheless, according to our numerical experiences it is an efficient approximation in many situations.

$\mathbb{R}^{d_2} \to \mathbb{R}$ be chosen in the respective spaces. Let $\mathcal{F}^{\mathbf{u}}$ and $\mathcal{F}^{\mathbf{v}}$ denote the reproducing kernel Hilbert spaces (RKHS) (Aronszajn, 1950; Wahba, 1999; Schölkopf et al., 1999) associated with the kernels. Here, $\mathcal{F}^{\mathbf{u}}$ and $\mathcal{F}^{\mathbf{v}}$ are function spaces having elements that perform mappings $\mathbb{R}^{d_1} \to \mathbb{R}$ and $\mathbb{R}^{d_2} \to \mathbb{R}$, respectively. Then the mutual dependence between $\mathbf{u}$ and $\mathbf{v}$ can be measured, for instance, by the following expression:

$$J^*_{\text{KCCA}}(\mathbf{u},\mathbf{v},\mathcal{F}^{\mathbf{u}},\mathcal{F}^{\mathbf{v}}) := \sup_{g\in\mathcal{F}^{\mathbf{u}}, h\in\mathcal{F}^{\mathbf{v}}} \operatorname{corr}[g(\mathbf{u}),h(\mathbf{v})],$$

where *corr* denotes correlation.

The value of J^*_{KCCA} can be estimated empirically: assume that we have T samples both from $\mathbf{u}$ and from $\mathbf{v}$. These samples are $\mathbf{u}_1,\ldots,\mathbf{u}_T \in \mathbb{R}^{d_1}$ and $\mathbf{v}_1,\ldots,\mathbf{v}_T \in \mathbb{R}^{d_2}$. Then, using notations $\bar{g} := \frac{1}{T}\sum_{k=1}^{T} g(\mathbf{u}_k)$, $\bar{h} := \frac{1}{T}\sum_{k=1}^{T} g(\mathbf{v}_k)$, the empirical estimation of J^*_{KCCA} could be the following:

$$J^{*,emp}_{\text{KCCA}}(\mathbf{u},\mathbf{v},\mathcal{F}^{\mathbf{u}},\mathcal{F}^{\mathbf{v}}) := \sup_{g\in\mathcal{F}^{\mathbf{u}}, h\in\mathcal{F}^{\mathbf{v}}} \frac{\frac{1}{T}\sum_{t=1}^{T}[g(\mathbf{u}_t)-\bar{g}][h(\mathbf{v}_t)-\bar{h}]}{\sqrt{\frac{1}{T}\sum_{t=1}^{T}[g(\mathbf{u}_t)-\bar{g}]^2}\sqrt{\frac{1}{T}\sum_{t=1}^{T}[h(\mathbf{v}_t)-\bar{h}]^2}}.$$

However, it is worth including some regularization for J^*_{KCCA} (Fukumizu et al., 2007), therefore J^*_{KCCA} is modified to

$$J_{\text{KCCA}}(\mathbf{u},\mathbf{v},\mathcal{F}^{\mathbf{u}},\mathcal{F}^{\mathbf{v}}) := \sup_{g\in\mathcal{F}^{\mathbf{u}}, h\in\mathcal{F}^{\mathbf{v}}} \frac{\operatorname{cov}[g(\mathbf{u}),h(\mathbf{v})]}{\sqrt{\operatorname{var}[g(\mathbf{u})]+\kappa\|g\|^2_{\mathcal{F}^{\mathbf{u}}}}\sqrt{\operatorname{var}[h(\mathbf{v})]+\kappa\|h\|^2_{\mathcal{F}^{\mathbf{v}}}}}, \tag{11}$$

where expression 'var' stands for variance, $\kappa > 0$ is the regularization parameter, $\|.\|^2_{\mathcal{F}^{\mathbf{u}}}$ and $\|.\|^2_{\mathcal{F}^{\mathbf{v}}}$ denote the RKHS norm of their arguments in $\mathcal{F}^{\mathbf{u}}$ and $\mathcal{F}^{\mathbf{v}}$, respectively. Now, expanding the denominator up to second order in κ, setting the expectation value of the samples to zero in the respective RKHSs, and using the notation $\kappa_2 := \frac{\kappa T}{2}$ (Bach and Jordan, 2002), the empirical estimation of (11) is

$$\hat{J}^{emp}_{\text{KCCA}}(\mathbf{u},\mathbf{v},\mathcal{F}^{\mathbf{u}},\mathcal{F}^{\mathbf{v}}) = \sup_{\mathbf{c}_1\in\mathbb{R}^T, \mathbf{c}_2\in\mathbb{R}^T} \frac{\mathbf{c}_1^*\widetilde{\mathbf{K}}^{\mathbf{u}}\widetilde{\mathbf{K}}^{\mathbf{v}}\mathbf{c}_2}{\sqrt{\mathbf{c}_1^*\left(\widetilde{\mathbf{K}}^{\mathbf{u}}+\kappa_2\mathbf{I}_T\right)^2\mathbf{c}_1}\sqrt{\mathbf{c}_2^*\left(\widetilde{\mathbf{K}}^{\mathbf{v}}+\kappa_2\mathbf{I}_T\right)^2\mathbf{c}_2}}, \tag{12}$$

where $\widetilde{\mathbf{K}}^{\mathbf{u}}, \widetilde{\mathbf{K}}^{\mathbf{v}} \in \mathbb{R}^{T\times T}$ are the so-called centered kernel matrices: These matrices are derived from kernel matrices $\mathbf{K}^{\mathbf{u}} = [k(\mathbf{u}_i,\mathbf{u}_j)]_{i,j=1,\ldots,T}, \mathbf{K}^{\mathbf{v}} = [k(\mathbf{v}_i,\mathbf{v}_j)]_{i,j=1,\ldots,T} \in \mathbb{R}^{T\times T}$, as is described below. Let $\mathbf{1}_T \in \mathbb{R}^T$ denote a vector whose all elements are equal to 1 and let $\mathbf{H} := \mathbf{I}_T - \frac{1}{T}\mathbf{1}_T\mathbf{1}_T^* \in \mathbb{R}^{T\times T}$ denote the so-called T-dimensional centering matrix. Then $\widetilde{\mathbf{K}}^{\mathbf{u}} := \mathbf{H}\mathbf{K}^{\mathbf{u}}\mathbf{H}$, $\widetilde{\mathbf{K}}^{\mathbf{v}} := \mathbf{H}\mathbf{K}^{\mathbf{v}}\mathbf{H}$.

Computing the stationary points of $\hat{J}^{emp}_{\text{KCCA}}$ in (12), that is, setting $\mathbf{0} = \frac{\partial \hat{J}^{emp}_{\text{KCCA}}}{\partial \mathbf{c}}$, the resulting task is to solve a *generalized eigenvalue problem* of the form $\mathbf{C}\boldsymbol{\xi} = \mu\mathbf{D}\boldsymbol{\xi}$:

$$\begin{pmatrix} (\widetilde{\mathbf{K}}^{\mathbf{u}}+\kappa_2\mathbf{I}_T)^2 & \widetilde{\mathbf{K}}^{\mathbf{u}}\widetilde{\mathbf{K}}^{\mathbf{v}} \\ \widetilde{\mathbf{K}}^{\mathbf{v}}\widetilde{\mathbf{K}}^{\mathbf{u}} & (\widetilde{\mathbf{K}}^{\mathbf{v}}+\kappa_2\mathbf{I}_T)^2 \end{pmatrix} \begin{pmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{pmatrix} = (1+\gamma) \begin{pmatrix} (\widetilde{\mathbf{K}}^{\mathbf{u}}+\kappa_2\mathbf{I}_T)^2 & \mathbf{0} \\ \mathbf{0} & (\widetilde{\mathbf{K}}^{\mathbf{v}}+\kappa_2\mathbf{I}_T)^2 \end{pmatrix} \begin{pmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{pmatrix},$$

where the objective is to maximize $\gamma := \mathbf{c}_1^*\widetilde{\mathbf{K}}^{\mathbf{u}}\widetilde{\mathbf{K}}^{\mathbf{v}}\mathbf{c}_2$. Our task is to estimate $\hat{J}^{emp}_{\text{KCCA}}$, the maximum of γ.

Generalization for many variables The KCCA method can be generalized for more than two random variables and can be used to measure pair-wise dependence: Let us introduce the following notations: Let $\mathbf{y}^1 \in \mathbb{R}^{d_1}, \ldots, \mathbf{y}^M \in \mathbb{R}^{d_M}$ be random variables. We want to measure the dependence between these variables. Let positive semi-definite kernels $k^m(\cdot,\cdot) : \mathbb{R}^{d_m} \times \mathbb{R}^{d_m} \to \mathbb{R}$ $(m = 1, \ldots, M)$ be chosen in the respective spaces. Let $\mathcal{F}^m$ denote the RKHS associated with kernel $k^m(\cdot,\cdot)$. Having T samples $\mathbf{y}_1^m, \ldots, \mathbf{y}_T^m$ for all random variables $\mathbf{y}^m$ $(m = 1, \ldots, M)$, matrices $\mathbf{K}^m := [k^m(\mathbf{y}_i^m, \mathbf{y}_j^m)]_{i,j=1,\ldots,T} \in \mathbb{R}^{T\times T}$ and $\widetilde{\mathbf{K}}^m := \mathbf{H}\mathbf{K}^m\mathbf{H} \in \mathbb{R}^{T\times T}$ can be created. Let the regularization parameter be chosen as $\kappa > 0$ and let κ_2 denote the auxiliary variable $\kappa_2 := \frac{\kappa T}{2}$. It can be proven that the computation of $\hat{J}^{emp}_{\text{KCCA}}$ involves the solution of the following generalized eigenvalue problem:

$$\begin{pmatrix} (\widetilde{\mathbf{K}}^1 + \kappa_2 \mathbf{I}_T)^2 & \widetilde{\mathbf{K}}^1\widetilde{\mathbf{K}}^2 & \cdots & \widetilde{\mathbf{K}}^1\widetilde{\mathbf{K}}^M \\ \widetilde{\mathbf{K}}^2\widetilde{\mathbf{K}}^1 & (\widetilde{\mathbf{K}}^2 + \kappa_2 \mathbf{I}_T)^2 & \cdots & \widetilde{\mathbf{K}}^2\widetilde{\mathbf{K}}^M \\ \vdots & \vdots & & \vdots \\ \widetilde{\mathbf{K}}^M\widetilde{\mathbf{K}}^1 & \widetilde{\mathbf{K}}^M\widetilde{\mathbf{K}}^2 & \cdots & (\widetilde{\mathbf{K}}^M + \kappa_2 \mathbf{I}_T)^2 \end{pmatrix} \begin{pmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \\ \vdots \\ \mathbf{c}_M \end{pmatrix} = \tag{13}$$

$$= \lambda \begin{pmatrix} (\widetilde{\mathbf{K}}^1 + \kappa_2 \mathbf{I}_T)^2 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & (\widetilde{\mathbf{K}}^2 + \kappa_2 \mathbf{I})^2 & \cdots & \mathbf{0} \\ \vdots & \vdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & (\widetilde{\mathbf{K}}^M + \kappa_2 \mathbf{I})^2 \end{pmatrix} \begin{pmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \\ \vdots \\ \mathbf{c}_M \end{pmatrix}.$$

Analogously to the two-variable-case, the largest eigenvalue of this task is a measure of the value of the pair-wise dependence of the random variables.

The KGV Method Equation (2) in Section 2.3 indicates that the ISA task can be seen as the minimization of the mutual information. The basic idea of the KGV technique is that—even for non-Gaussian variables—it estimates the mutual information by the Gaussian approximation (Bach and Jordan, 2002). Namely, let $\mathbf{y} = [\mathbf{y}^1; \ldots, \mathbf{y}^M]$ be multidimensional normal random variable with covariance matrix $\mathbf{C}$. Let $\mathbf{C}^{i,j} \in \mathbb{R}^{d_m\times d_m}$ denote the cross-covariance between components of $\mathbf{y}^m \in \mathbb{R}^{d_m}$. The mutual information between components $\mathbf{y}^1, \ldots, \mathbf{y}^M$ is (Cover and Thomas, 1991):

$$I(\mathbf{y}^1, \ldots, \mathbf{y}^M) = -\frac{1}{2} \log \left(\frac{\det \mathbf{C}}{\prod_{m=1}^M \det \mathbf{C}^{m,m}} \right).$$

The quotient $\frac{\det \mathbf{C}}{\prod_{m=1}^M \det \mathbf{C}^{m,m}}$ is called the *generalized variance*. If $\mathbf{y}$ is *not normal*—this is the typical situation in the ISA task—then let us transform the individual components $\mathbf{y}^m$ using feature mapping φ associated with the reproducing kernel and assume that the image is a normal variable. Thus, the cost function

$$J_{\text{KGV}}(\mathbf{W}) := -\frac{1}{2} \log \left[\frac{\det(\mathcal{K})}{\prod_{m=1}^M \det(\mathcal{K}^{m,m})} \right] \tag{14}$$

is associated with the ISA task. In Equation (14) $\phi(\mathbf{y}) := [\varphi(\mathbf{y}^1); \ldots; \varphi(\mathbf{y}^M)]$, $\mathcal{K} := cov[\phi(\mathbf{y})]$, and the sub-matrices are $\mathcal{K}^{i,j} = cov[\varphi(\mathbf{y}^i), \varphi(\mathbf{y}^j)]$. Expression $\frac{\det(\mathcal{K})}{\prod_{m=1}^M \det(\mathcal{K}^{m,m})}$ is called the *kernel generalized variance* (KGV).

The next theorem shows that the KGV technique can be interpreted as a decorrelation based method:

Theorem 2 *Let $\Sigma \in \mathbb{R}^{D\times D}$ be a positive semi-definite matrix, let $\Sigma^{m,m} \in \mathbb{R}^{d_m\times d_m}$ denote the m^{th} block in the diagonal of matrix Σ, and let $D = \sum_{m=1}^{M} d_m$. Then the function*

$$0 \leq Q(\Sigma) := -\frac{1}{2}\log\left[\frac{\det(\Sigma)}{\prod_{m=1}^{M}\det(\Sigma^{m,m})}\right]$$

is 0 iff $\Sigma = blockdiag(\Sigma^{1,1},\ldots,\Sigma^{M,M})$.

This theorem can be proven for $d_m \geq 1$, as in the case of $d_m = 1$ (Cover and Thomas, 1991), see the work of Szabó and Lőrincz (2006). The theorem implies the following:

Corollary *Setting $\Sigma := \mathcal{K}$, the KGV technique is a decorrelation technique according to feature mapping φ. The KGV technique aims at minimizing of cross-covariances $\mathcal{K}^{i,j} = cov[\varphi(\mathbf{y}^i),\varphi(\mathbf{y}^j)]$ to* $\mathbf{0}$.

We note that the kernel covariance (KC) ICA method (Gretton et al., 2005)—similarly to the KCCA method—can be extended to measure the mutual dependence of multidimensional random variables and thus to solve the ISA task. Again, the computation of the cost function can be converted to the solution of a generalized eigenvalue problem. This eigenvalue problem is provided in Appendix B for the sake of completeness.

Note 4 *The KCCA, KGV and KC methods can estimate only* pair-wise *dependence. Nonetheless, the joint mutual information can be estimated by recursive methods computing pair-wise mutual information: for the mutual information of random variables $\mathbf{y}^m \in \mathbb{R}^{d_m}$ $(m = 1,\ldots,M)$ it can be shown that the recursive relation*

$$I(\mathbf{y}^1,\ldots,\mathbf{y}^M) = \sum_{m=1}^{M} I\left(\mathbf{y}^m, \left[\mathbf{y}^{m+1},\ldots,\mathbf{y}^M\right]\right) \tag{15}$$

holds (Cover and Thomas, 1991). Thus, for example, the KCCA eigenvalue problem of (13) *can be replaced by pair-wise estimation of mutual information. We note that the tree-dependent component analysis model (Bach and Jordan, 2003) estimates the joint mutual information from the pair-wise mutual information.*

4.2 Optimization of ISA Costs

There are several possibilities to optimize ISA cost functions:

1. Without ICA preprocessing, optimization problems concern either the *Stiefel manifold* (Edelman et al., 1998; Lippert, 1998; Plumbley, 2004; Quinquis et al., 2006) or the *flag manifold* (Nishimori et al., 2006). According to our experiences, these gradient based optimization methods may be stuck in poor local minima.

2. According to the ISA Separation Theorem, it may be sufficient to search for optimal permutation of the ICA components provided by ICA preprocessing. We applied greedy permutation search: two coordinates of different subspaces are exchanged provided that this change decreases cost function J. Here, J denotes, for example, J_{JFD}, J_{KCCA}, or J_{KGV} depending on the ISA technique applied. The variable of J is the permutation matrix $\mathbf{P}$ using the parametrization $\mathbf{W}_{\text{ISA}} = \mathbf{P}\mathbf{W}_{\text{ICA}}$. Pseudocode is provided in Table 2. Our experiences show that greedy

```
Input of the algorithm
    ISA observation: ${\mathbf{x}(t)}_{t=1,...,T}$
Optimization[9]
    ICA: on the whitened observation $\mathbf{x} \Rightarrow \hat{\mathbf{s}}_{\text{ICA}}$ estimation
    Permutation search
        $\mathbf{P} := \mathbf{I}_D$
        repeat
            sequentially for $\forall p \in \mathcal{G}^{m_1}, q \in \mathcal{G}^{m_2} (m_1 \neq m_2)$ :
                if $J(\mathbf{P}_{pq}\mathbf{P}) < J(\mathbf{P})$
                    $\mathbf{P} := \mathbf{P}_{pq}\mathbf{P}$
                end
        until $J(\cdot)$ decreases in the sweep above
Estimation
    $\hat{\mathbf{s}}_{\text{ISA}} = \mathbf{P}\hat{\mathbf{s}}_{\text{ICA}}$
```

Table 2: Pseudocode of the ISA Algorithm. Cost J stands for the ISA cost function of JFD, KCCA, or KGV methods. The permutation matrix of the ISA Separation Theorem is the variable of J.

permutation search is often sufficient for the estimation of the ISA subspaces. However, it is easy to generate examples in which this is not true (Póczos and Lőrincz, 2005a). In such cases, global permutation search method of higher computational burden may become necessary (Szabó et al., 2006a).

4.3 Different and Unknown Component Dimensions

Here we give a quick overview how one can handle situations when the dimensions of the subspaces are unequal, unknown, or both. Note that the introduced uBSSD-ISA reduction, the ISA ambiguities (Theis, 2006) and the Separation Theorem remain the same for subspaces of different dimensions, and thus it is sufficient to consider the ISA problem.

1. If the dimensions of the subspaces are different but known, the ISA task can be solved
   (a) the mask $\mathbf{N}$ of the JFD method should be modified (see Equation 10).
   (b) the kernel-ISA methods include this situation; they were introduced for different subspace dimensions.
2. If the dimension of the hidden source $\mathbf{s}$ is known, but the individual dimensions of components $\mathbf{s}^m$ are not, then clustering can exploit the dependencies between the coordinates of the estimated sources. For example:
   (a) If we assume that the hidden $\mathbf{s}$ source has block diagonal AR dynamics of the form $\mathbf{s}(t+1) = \mathbf{F}\mathbf{s}(t) + \mathbf{e}(t)$—$\mathbf{F} = blockdiag\left(\mathbf{F}^1, \ldots, \mathbf{F}^M\right)$—then connectivity of the estimated matrix $\hat{\mathbf{F}}$ may help (Póczos and Lőrincz, 2006). One may assume that the i and the j coordinates are '$\hat{\mathbf{F}}$-connected' if value $\max\{|\hat{F}_{ij}|, |\hat{F}_{ji}|\}$ is above a certain threshold.

9. Let $\mathcal{G}^1, \ldots, \mathcal{G}^M$ denote the indices of the $1^{st}, \ldots, M^{th}$ subspaces, that is, $\mathcal{G}^m := \{(m-1)d+1, \ldots, md\}$, and permutation matrix $\mathbf{P}_{pq}$ exchanges coordinates p and q.

(b) Similar considerations can be applied in the ISA problem, for example, by using cumulant based matrices (Theis, 2006).

(c) Weaknesses of the threshold based method include (i) the uncertainty in choosing the threshold, and (ii) the fact that the methods are sensitive to the threshold. More robust solutions can be designed if the dependencies, for example, the mutual information amongst the coordinates, are used to construct an adjacency matrix and apply a clustering method for this matrix. One might use, for example, hierarchical (Stögbauer et al., 2004) or tree-structured clustering methods (Bach and Jordan, 2003).

5. Illustrations

The efficiency of the algorithms of Section 4 are illustrated. Test cases are introduced in Section 5.1. The quality of the solutions will be measured by the normalized Amari-error, the Amari-index (Section 5.2). Numerical results are presented in Section 5.3.

5.1 Databases

We define five databases to study our identification algorithms. We do not know whether they satisfy (8) or not. According to our experiences, the ISA Separation Theorem works on these examples.

5.1.1 THE 3D-GEOM, THE CELEBRITIES AND THE ABC DATABASE

The first 3 databases are illustrated in Figure 1. In the *3D-geom* test $\mathbf{s}^m$s were random variables uniformly distributed on 3-dimensional geometric forms ($d = 3$). We chose 6 different components ($M = 6$) and, as a result, the dimension of the hidden source $\mathbf{s}$ is $D_s = 18$. The *celebrities* test has 2-dimensional source components generated from cartoons of celebrities ($d = 2$).[10] Sources $\mathbf{s}^m$ were generated by sampling 2-dimensional coordinates proportional to the corresponding pixel intensities. In other words, 2-dimensional images of celebrities were considered as density functions. $M = 10$ was chosen. In the *ABC* database, hidden sources $\mathbf{s}^m$ were uniform distributions defined by 2-dimensional images ($d = 2$) of the English alphabet. The number of components varied as $M = 2, 5, 10, 15$, and thus the dimension of the source D_s was $4, 10, 20, 30$, respectively.

5.1.2 THE ALL-K-INDEPENDENT DATABASE

The d-dimensional hidden components $\mathbf{u} := \mathbf{s}^m$ were created as follows: coordinates $u_i(t)$ $(i = 1,\ldots,k)$ were uniform random variables on the set $\{0,\ldots,\text{k-1}\}$, whereas u_{k+1} was set to $mod(u_1 + \ldots + u_k, k)$. In this construction, every k-element subset of $\{u_1,\ldots,u_{k+1}\}$ is made of independent variables. This database is called the *all-k-independent* problem (Póczos and Lőrincz, 2005a; Szabó et al., 2006a). In our simulations $d = k + 1$ was set to 3 or 4 and we used 2 components ($M = 2$). Thus, source dimension D_s was either 6 or 8.

5.1.3 THE BEATLES DATABASE

Our *Beatles* test is a non-i.i.d. example. Here, hidden sources are stereo Beatles songs.[11] 8 kHz sampled portions of two songs (A Hard Day's Night, Can't Buy Me Love) made the hidden $\mathbf{s}^m$s.

10. See http://www.smileyworld.com.
11. See http://rock.mididb.com/beatles/.

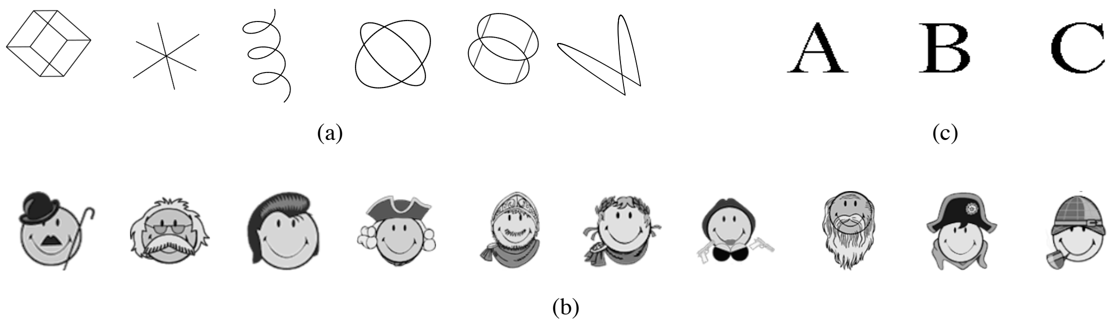


Figure 1: Illustration of the *3D-geom*, *celebrities* and *ABC* databases. (a): database *3D-geom*, 6 3-dimensional components ($M = 6, d = 3$). Hidden sources are uniformly distributed variables on 3-dimensional geometric objects. (b): database *celebrities*. Density functions of the hidden sources are proportional to the pixel intensities of the 2-dimensional images ($d = 2$). Number of hidden components: $M = 10$. (c): database *ABC*. Here, the hidden sources $\mathbf{s}^m$ are uniformly distributed on images ($d = 2$) of letters. Number of components M varies between 2 (A and B) and 15 (A-O).

Thus, the dimension of the components d was 2, the number of the components M was 2, and the dimension of the problem D_s was 4.

5.2 Normalized Amari-error, the Amari-index

We have shown in Section 3.1 that the uBSSD task can be reduced to an ISA task. Consequently, we use ISA performance measure to evaluate our algorithms. Assume that there are M pieces of d-dimensional hidden components in the ISA task, $\mathbf{A}$ is the mixing matrix, and $\mathbf{W}$ is the estimated demixing matrix. Then optimal estimation provides matrix $\mathbf{G} := \mathbf{WA}$, a block-permutation matrix made of $d \times d$ sized blocks. Let matrix $\mathbf{G} \in \mathbb{R}^{D \times D}$ be decomposed into $d \times d$ blocks: $\mathbf{G} = \left[\mathbf{G}^{ij}\right]_{i,j=1,\ldots,M}$. Let $g^{i,j}$ denote the sum of the absolute values of the elements of matrix $\mathbf{G}^{i,j} \in \mathbb{R}^{d \times d}$. We used the normalized version (Szabó et al., 2006a) of the Amari-error (Amari et al., 1996) adapted to the ISA task (Theis, 2005a,b) defined as:

$$r(\mathbf{G}) := \frac{1}{2M(M-1)} \left[\sum_{i=1}^{M} \left(\frac{\sum_{j=1}^{M} g^{ij}}{\max_j g^{ij}} - 1 \right) + \sum_{j=1}^{M} \left(\frac{\sum_{i=1}^{M} g^{ij}}{\max_i g^{ij}} - 1 \right) \right].$$

We refer to the normalized Amari-error as the Amari-index. One can see that $0 \leq r(\mathbf{G}) \leq 1$ for any matrix $\mathbf{G}$, and $r(\mathbf{G}) = 0$ if and only if $\mathbf{G}$ is a block-permutation matrix with $d \times d$ sized blocks. Normalization is advantageous; we can compare the precision of ISA procedures and procedures, which are reduced to ISA tasks.

5.3 Simulations

Results on databases *3D-geom*, *celebrities*, *ABC*, *all-k-independent* and *Beatles* are provided here. These experimental studies have two main parts:

1. The efficiency of the JFD method on the uBSSD task is demonstrated in Section 5.3.1.

2. The derived KCCA, KGV kernel-ISA methods were tested on ISA tasks. We show examples in which these methods are favorable over the JFD method in Section 5.3.2.

In both cases the tasks are either ISA tasks or can be reduced to ISA (Section 3.1). Thus, we used the Amari-index (Section 5.2) to measure and compare the performance of the different methods. For each individual parameter, 50 random runs were averaged. Our parameters are: T, the sample number of observations $\mathbf{x}(t)$, L, the parameter of the length of the convolution (the length of the convolution is $L+1$), M, the number of the components, and d, the dimension of the components, depending on the test. Random run means random choice of quantities $\mathbf{H}[z]$ and $\mathbf{s}$.

5.3.1 JFD ON uBSSD

Our results concerning the uBSSD task are delineated. As we showed in Section 3.1, the temporal concatenation can turn the uBSSD task into an ISA problem. These ISA tasks associated with simple uBSSD problems can easily become more than 100-dimensional. Earlier ISA methods cannot deal with such 'high dimensional' problems. This is why we resorted to the recent JFD method (Section 4.1.1), which seemed to be efficient in solving such large problems under the following circumstances: Equation (7) implies that the dimension D_{ISA} of the derived ISA task with fixed L and D_s decreases provided that the difference $D_x - D_s \geq 1$ increases. This coincides with our experiences: the higher this difference is, the smaller number of samples can reach the same precision. Below, studies for $D_x - D_s = D_s$ ($D_x = 2D_s$) are presented. This choice was amenable to the JFD method.[12] In this case the dimension of the ISA task in (7) simplifies to the form

$$D_{\text{ISA}} = 2D_sL.$$

The JFD technique works with the pseudocode given in Table 2: it reduces the uBSSD task to the ISA task, where the fastICA algorithm (Hyvärinen and Oja, 1997) was chosen to perform the ICA computation. In the JFD cost, we chose manifold $\mathcal{F}$ as $\mathcal{F} := \{\mathbf{u} \to \cos(\mathbf{u}), \mathbf{u} \to \cos(2\mathbf{u})\}$, where the functions operated on the coordinates separately (Szabó and Lőrincz, 2006). For the 'observations', the elements of mixing matrices $\mathbf{H}_l$ in Equation (1) were generated independently from standard normal distributions.[13]

	$L=1$	$L=2$	$L=3$	$L=4$	$L=5$
3D-geom	0.25% ($\pm$0.01)	0.27% ($\pm$0.03)	0.28% ($\pm$0.02)	0.29% ($\pm$0.03)	0.29% ($\pm$0.01)
celebrities	0.37% ($\pm$0.01)	0.38% ($\pm$0.01)	0.39% ($\pm$0.01)	0.39% ($\pm$0.01)	0.40% ($\pm$0.01)

Table 3: The Amari-index of the JFD method for database *3D-geom* and *celebrities*, for different convolution lengths: average $\pm$ deviation. Number of samples: $T = 100{,}000$. For other sample numbers between $1{,}000 \leq T < 100{,}000$ see Figure 2.

We studied the dependence of the precision versus the sample number on databases *3D-geom* and *celebrities*. The dimension and the number of the components were $d = 3$ and $M = 6$ for the *3D-geom* database and $d = 2$ and $M = 10$ for the *celebrities* database, respectively. In both cases the sample number T varied between $1{,}000$ and $100{,}000$. The parameter of the length of the

12. We note that the hardest $D_x - D_s = 1$ task is also feasible. However, the sample number necessary to find the solution grows considerably, as can be expected from (5).
13. Uniform distribution on $[0.1]$, instead of normal distribution, showed similar performance.

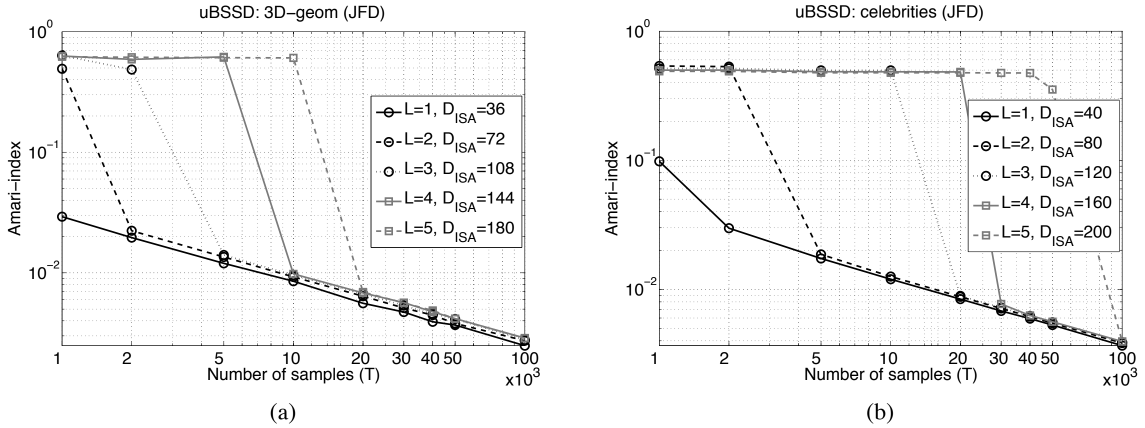


Figure 2: Estimation error of the JFD method on the *3D-geom* and *celebrities* databases: Amari-index as a function of sample number on log-log scale for different convolution lengths. (a): *3D-geom*, (b): *celebrities* database. Dimension of the ISA task: D_{ISA}. For further information, see Table 3.

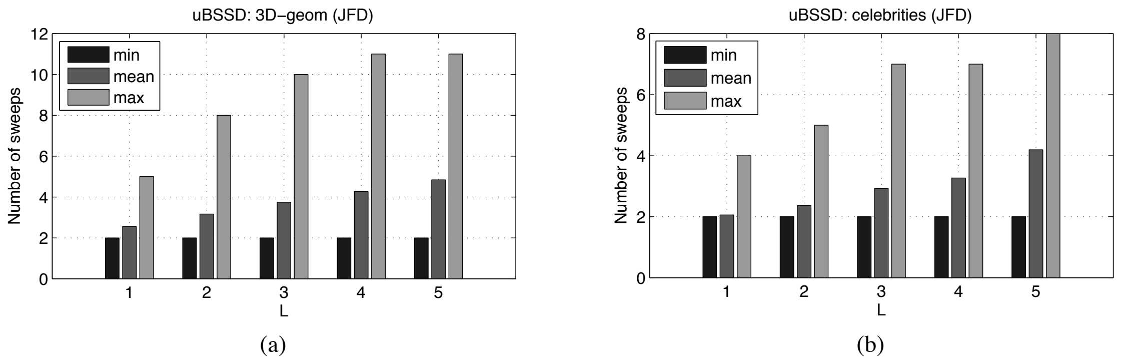


Figure 3: Number of sweeps in permutation search needed for the JFD method as a function of the convolution length. (a): *3D-geom*, (b): celebrities database. Black: minimum, gray: average, light gray: maximum.

convolution took $L = 1, \ldots, 5$ values. Thus, the length of the convolution changed between 2 and 6. Our results are summarized in Figure 2. The values of the errors are given in Table 3. The number of sweeps needed to optimize the permutations after performing ICA is provided in Figure 3. Figures 4 and 5 illustrate the estimations of the JFD technique on the *3D-geom* and the *celebrities* databases, respectively.

Figure 2 demonstrates that the JFD algorithm was able to uncover the hidden components with high precision. The precision of the estimations shows similar characteristics on the *3D-geom* and the *celebrities* databases. The Amari-index is approximately constant for small sample numbers. For each curve, above a certain threshold the Amari-index decreases suddenly and after the sudden decrease the precision follows a power law $r(T) \propto T^{-c}$ $(c > 0)$. The power law decline is manifested by straight line on log-log scale. The slopes of these straight lines are very close to one another.

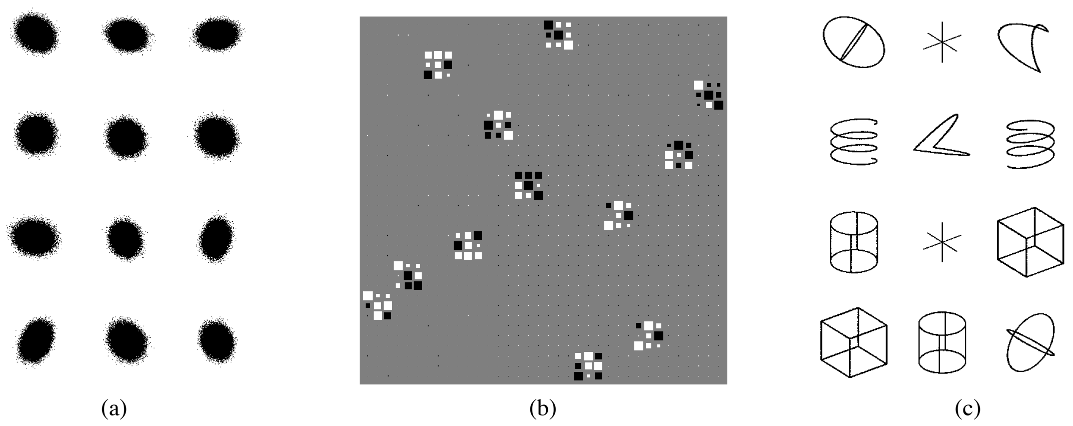


Figure 4: Illustration of the JFD method on the uBSSD task for the *3D-geom* database. Sample number $T = 100{,}000$, convolution length $L = 1$, Amari-index: 0.25%. (a): observed convolved signals $\mathbf{x}(t)$. (b) Hinton-diagram: the product of the mixing matrix of the derived ISA task and the estimated demixing matrix (= approximately block-permutation matrix with 3×3 blocks). (c): estimated components. Note: hidden components are recovered $L + L' = 2$ times, up to permutation and orthogonal transformation.

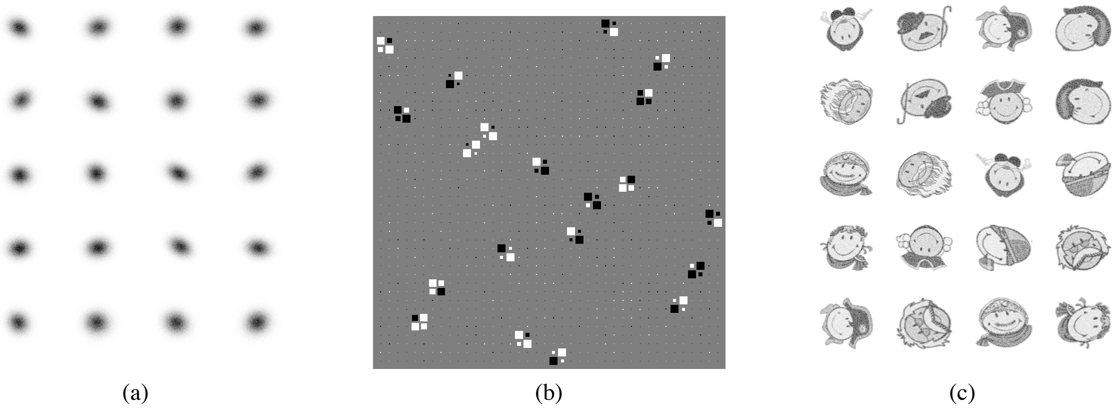
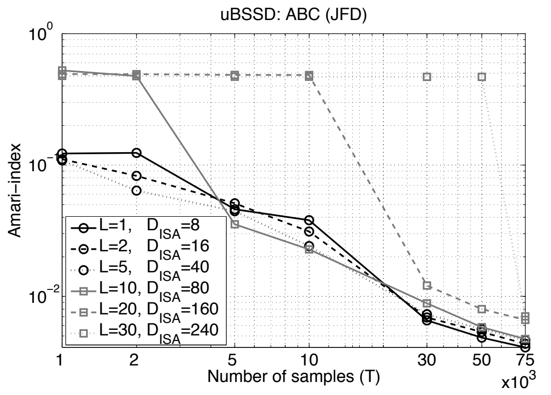
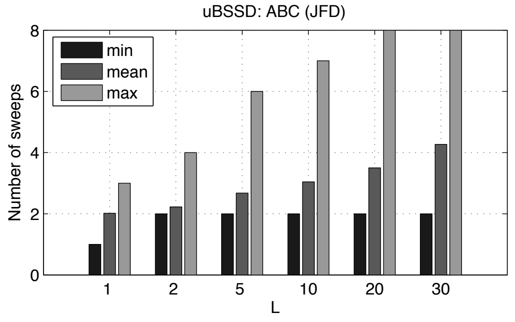


Figure 5: Illustration of the JFD method on the uBSSD task for the *celebrities* database. Sample number $T = 100{,}000$, convolution length $L = 1$, Amari-index: 0.37%. (a): observed convolved signals $\mathbf{x}(t)$. (b) Hinton-diagram: the product of the mixing matrix of the derived ISA task and the estimated demixing matrix (= approximately block-permutation matrix with 2×2 blocks). (c): estimated components. Note: hidden components are recovered $L + L' = 2$ times, up to permutation and orthogonal transformation.

The number of sweeps was between 2 and 11 (2 and 8) for the *3D-geom* (*celebrities*) tests over all sample numbers, for $1 \leq L \leq 5$ and for 50 random initializations. According to Table 3, the Amari-index for sample number $T = 100{,}000$ is below 1% ($0.25 - 0.40\%$) with small standard deviations ($0.01 - 0.03$).



(a)



(b)

Figure 6: (a): Amari-index of the JFD method on the *ABC* database as a function of sample number and for different convolution lengths on log-log scale. (b): Number of sweeps of permutation optimization on the derived ISA task as a function of convolution length. Dimension of the ISA task: D_{ISA}. Black: minimum, gray: average, light gray: maximum. For further information, see Table 4.

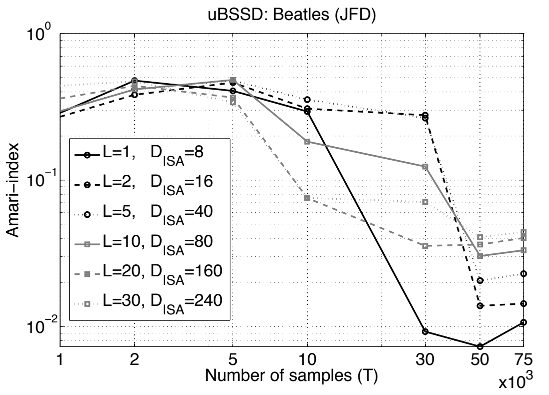
In another test the *ABC* database was used. The number and the dimensions of the components were minimal ($d = 2, M = 2$) and the dependence on the convolution length was tested. Parameter L took values on $1,2,5,10,20,30$. The number of observations varied between $1{,}000 \leq T \leq 75{,}000$. The Amari-index and the sweep number of the optimization are illustrated in Figure 6. Precise values of the Amari-index are provided in Table 4.

$L = 1$	$L = 2$	$L = 5$	$L = 10$	$L = 20$	$L = 30$
0.41% (±0.06)	0.44% (±0.05)	0.46% (±0.05)	0.47% (±0.03)	0.66% (±0.13)	0.70% (±0.11)

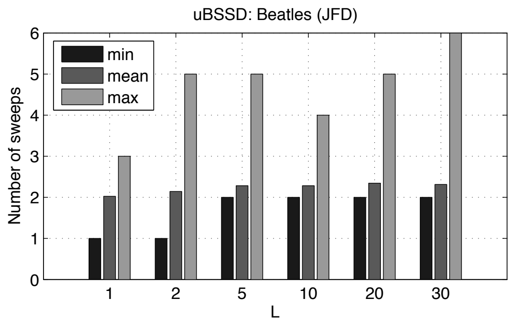
Table 4: Amari-index of the JFD method for *ABC* database for different convolution lengths: average ± deviation. Number of samples: $T = 75{,}000$. For other sample numbers between $1{,}000 \leq T < 75{,}000$ see Figure 6(a).

According to Figure 6, the JFD method found the hidden components. The 'power law' decline of the Amari-index, which was apparent in the *3D-geom* and the *celebrities* databases, appears for the *ABC* test, too. The figure indicates that for 75,000 samples and for $L = 30$ (convolution length is 31) the problem is still amenable to the JFD technique. The number of sweeps required for the optimization of the permutations was between 1 and 8 for all sample numbers $1{,}000 \leq T \leq 75{,}000$, parameters $1 \leq L \leq 30$ and for all 50 random initializations. According to Table 4, for sample number $T = 75{,}000$ the Amari-index stays below 1% on average ($0.41 - 0.7\%$) and has a small ($0.03 - 0.11$) standard deviation.

In the case of *Beatles* database, test parameters were similar to those of the *ABC* database: the number and the dimensions of the components were minimal ($d = 2$, $M = 2$) and the dependence on the convolution length was tested. Parameter L took values on $1,2,5,10,20,30$. The number of observations varied between $1{,}000 \leq T \leq 75{,}000$. The Amari-index and the sweep number of the



(a)



(b)

Figure 7: (a): Amari-index of the JFD method on the *Beatles* database as a function of sample number and for different convolution lengths on log-log scale. (b): Number of sweeps of permutation optimization on the derived ISA task as a function of convolution length. Dimension of the ISA task: D_{ISA}. Black: minimum, gray: average, light gray: maximum. For further information, see Table 5.

optimization are illustrated in Figure 7. Precise values of the Amari-index are provided in Table 5. The Hinton-diagrams are in Figure 8.

The Beatles songs are non-i.i.d sources and subsequent samples $\mathbf{s}^m$(t) and $\mathbf{s}^m(t-1)$ have dependencies. Thus, in $\mathbf{S}(t)$ the components of the (6) model that belong to the same song can not be distinguished. In the ideal case, however, the songs can be separated, that is, the separation of the 2 pieces of $(D_{\text{ISA}}/2)$-dimensional song-subspaces is possible. We measure the appearance of the 2 of $(D_{\text{ISA}}/2)$-dimensional blocks for the *Beatles* songs by means of the Amari-index. The results, which demonstrate the success of our method, are shown in Figure 7. It can be seen that the JFD method found the hidden components, for 50 and 75 thousand samples for $L = 30$ (convolution length is 31) too. The number of sweeps required for the optimization of the permutations was between 1 and 6 for all sample numbers $1{,}000 \leq T \leq 75{,}000$, parameters $1 \leq L \leq 30$ and for all 50 random initializations. According to Table 5, for sample number $T = 75{,}000$ the Amari-index is between 1.07% $(L = 1)$ and 4.43% $(L = 30)$ on the average and has a small $(0.04 - 0.09)$ standard deviation. Separation of the 2 of $D_{\text{ISA}}/2$ dimensional subspaces are illustrated in Figure 8 through the Hinton-diagrams.

$L = 1$	$L = 2$	$L = 5$	$L = 10$	$L = 20$	$L = 30$
1.07% (±0.04)	1.43% (±0.09)	2.29% (±0.07)	3.31% (±0.06)	4.03% (±0.06)	4.43% (±0.04)

Table 5: Amari-index of the JFD method for *Beatles* database for different convolution lengths: average ± deviation. Number of samples: $T = 75{,}000$. For other sample numbers between $1{,}000 \leq T < 75{,}000$ see Figure 7(a).

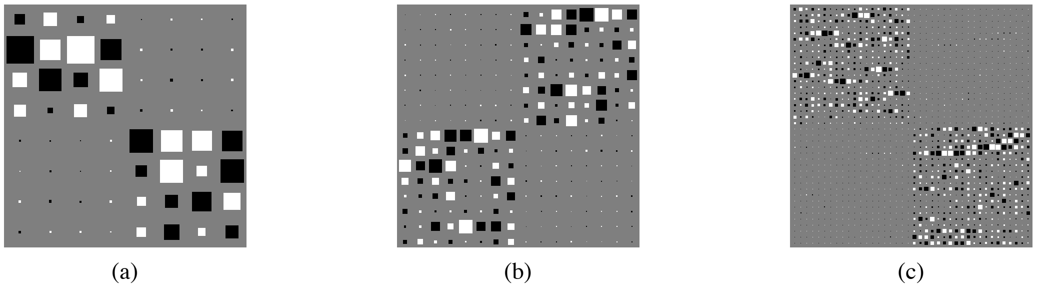


Figure 8: Hinton-diagrams of the JFD methods on the *Beatles* database for different convolution parameters. (a): $L = 1$ $(D_{\text{ISA}} = 8)$, (b): $L = 2$ $(D_{\text{ISA}} = 16)$, (c): $L = 5$ $(D_{\text{ISA}} = 40)$. In the ideal case, 2 pieces of $D_{\text{ISA}}/2$-dimensional blocks are formed by multiplying the mixing matrix of the derived ISA task and the estimated demixing matrix.

5.3.2 KERNEL-ISA TECHNIQUES

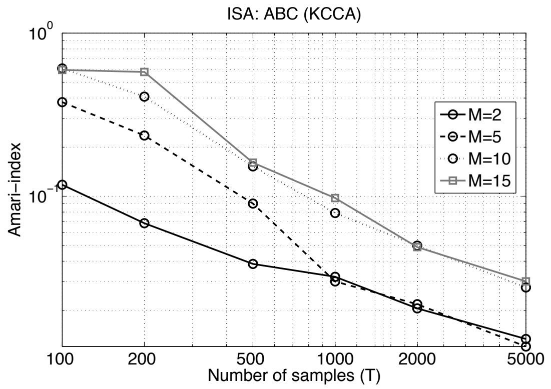
We study the efficiency of the KCCA and KGV kernel-ISA methods of Section 4.1.2. The kernel-ISA techniques was found to have a higher computational burden, but they also have advantages compared with the JFD technique for ISA tasks.

For the KCCA and KGV methods we also applied the pseudocode of Table 2. ICA was executed by the fastICA algorithm (Hyvärinen and Oja, 1997). The Gaussian kernel $k(\mathbf{u},\mathbf{v}) \propto \exp\left(\frac{-\|\mathbf{u}-\mathbf{v}\|^2}{2\sigma^2}\right)$ was chosen for both the KCCA and KGV methods and parameter σ was set to 5. In the KCCA method, regularization parameter $\kappa = 10^{-4}$ was applied. In the experiments, parameters (σ, κ) were proved to be reasonably robust. Mixing matrix $\mathbf{A}$ was generated randomly from the orthogonal group and the sample number was chosen from the interval $100 \leq T \leq 5{,}000$.

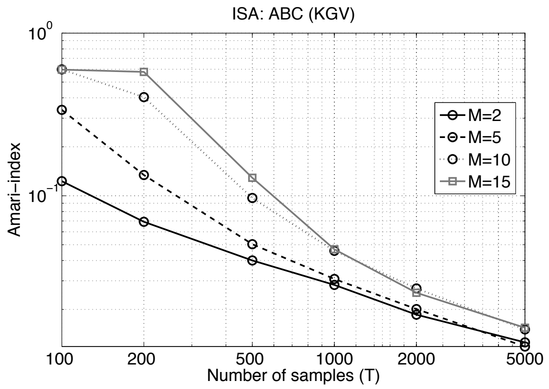
Our first ISA example concerns the *ABC* database. The dimension of a component was $d = 2$ and the number of the components M took different values ($M = 2, 5, 10, 15$). Precision of the estimations is shown in Figure 9, where the precision of the JFD method on the same database is also depicted. The number of sweeps required for the optimization of the permutations is shown in Figure 10 for different sample numbers and for different component numbers. The data are averaged over 50 random estimations. Figure 11 depicts the KCCA estimation for the *ABC* database.

Figure 9 shows that the KCCA and KGV kernel-ISA methods give rise to high precision estimations on the *ABC* database even for small sample numbers. The KGV method was more precise for all M values studied than the JFD method. The ratio of precisions could be as high as 4 (see sample number 500). The KCCA method is somewhat weaker. For smaller tasks ($M = 2$ and 5) and for small sample numbers it also exceeds the precision of the JFD method. Precision of the method become about the same for higher sample numbers and larger tasks. Sweep numbers of the KCCA (KGV) method were between 2 and 8 (2 and 6) (Figure 10). Note that one sweep is always necessary for our procedure (Table 2). A single sweep may be satisfactory if—by chance—the ICA provides the correct permutation.

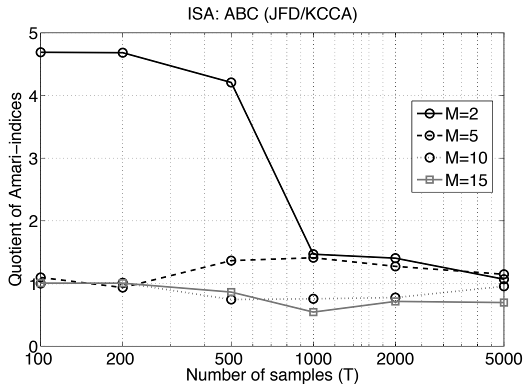
The other illustration concerns the *all-k-independent* database. This test can be difficult for ISA methods (Szabó et al., 2006a). Number of components M was 2. For $k = 2, 3$, when the dimension of the components $d = 3$ and 4, respectively, the KCCA and KGV kernel-ISA methods efficiently estimated the hidden components. The precision of the KCCA and KGV estimations



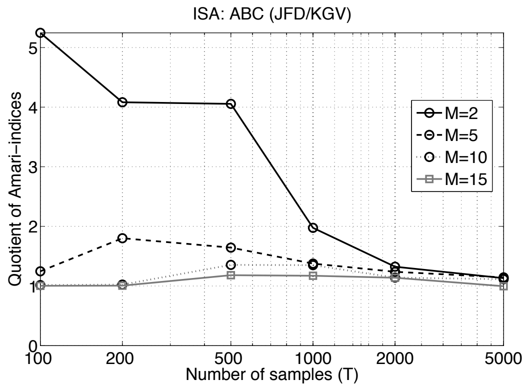
(a)



(b)



(c)



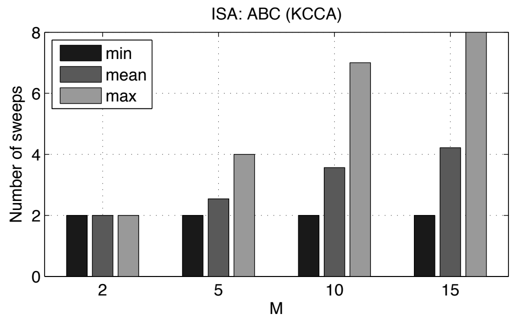
(d)

Figure 9: (a) and (b): Amari-index of the KCCA and KGV methods, respectively, for the *ABC* database as a function of sample number and for different numbers of components M. (c) and (d): Amari-index of the JFD method is divided by the Amari-index of the KCCA method and the KGV technique, respectively. For values larger (smaller) than 1 the kernel-ISA method is better (worse) than the JFD method.

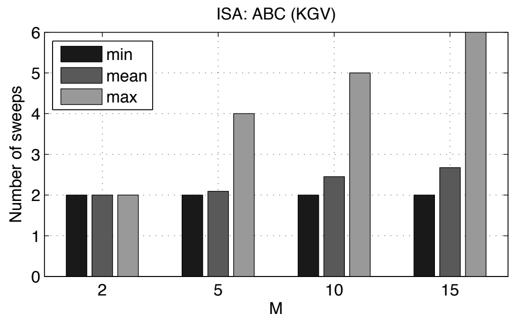
	$M = 2$	$M = 5$	$M = 10$	$M = 15$
KCCA	1.33% (±0.48)	1.20% (±0.17)	2.76% (±2.86)	3.00% (±2.21)
KGV	1.26% (±0.54)	1.18% (±0.17)	1.51% (±0.31)	1.54% (±0.34)

Table 6: Amari-index for the KCCA and the KGV methods for database *ABC*, for different component number M: average ± deviation. Number of samples: $T = 5{,}000$. For other sample numbers between $100 \leq T < 5{,}000$, see Figure 9.

as well as the comparison with the JFD method are shown in Figure 12. The average number of sweeps required for the optimization of the permutations for different k values and for 50 randomly initialized computations is provided in Figure 13. The values of the Amari-indices are shown in Table 7.

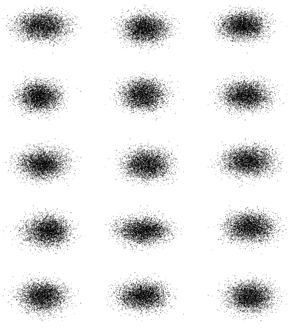


(a)

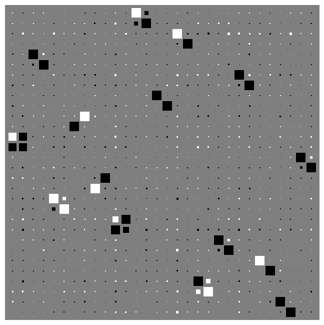


(b)

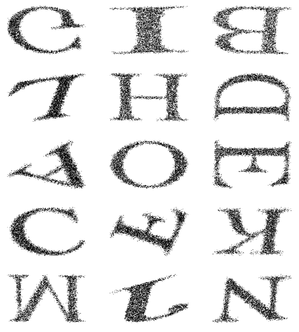
Figure 10: Number of sweeps for the KCCA and the KGV methods needed for the optimization of permutations as a function of component number M on the *ABC* database. (a): KCCA method, (b): KGV method. Black: minimum, gray: mean, light gray: maximum.



(a)



(b)



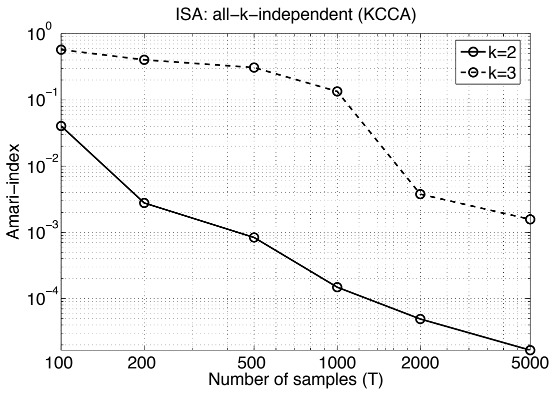
(c)

Figure 11: Illustration of the KCCA method for the *ABC* database. Sample number: $T = 5{,}000$. (a): observed mixed signals $\mathbf{x}(t)$. (b) Hinton-diagram: the product of the mixing matrix of the derived ISA task and the estimated demixing matrix (= approximately block-permutation matrix). (c): estimated components. Hidden components are recovered up to permutation and orthogonal transformation.

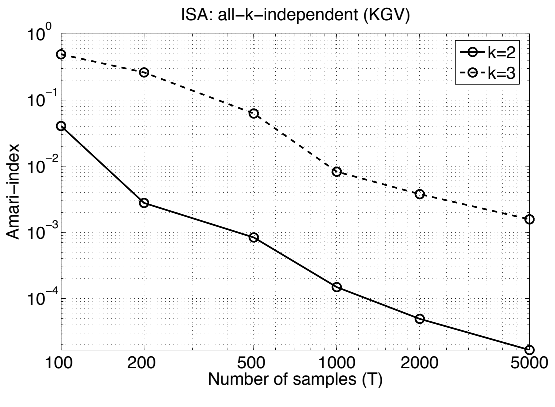
According to Figure 12 the two kernel-based methods exhibit similar precision. Both were superior to the JFD technique. The ratio of the Amari-indices for sample number 5,000 and for $k = 2$ is more than 15,000, for $k = 3$ it is more than 500. For details concerning the Amari-indices, see Table 7. These indices are close to each other for the KCCA and the KGV methods: 0.0017% for $k = 2$, 0.16% for $k = 3$ on average. Both kernel-ISA methods used $2-3$ sweeps for the optimization of the permutations (Figure 13).

6. Conclusions

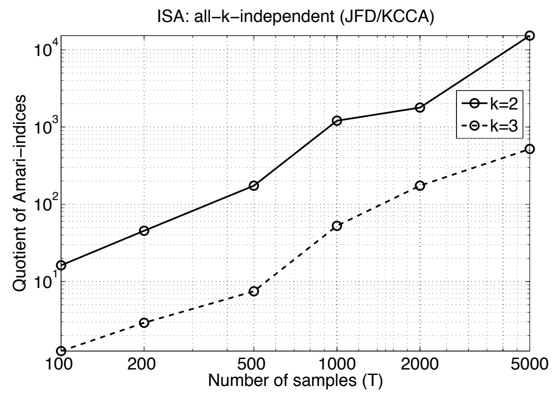
We have introduced a new model, the blind subspace deconvolution (BSSD) for data analysis. This model deals with the casual convolutive mixture of multidimensional independent sources. The



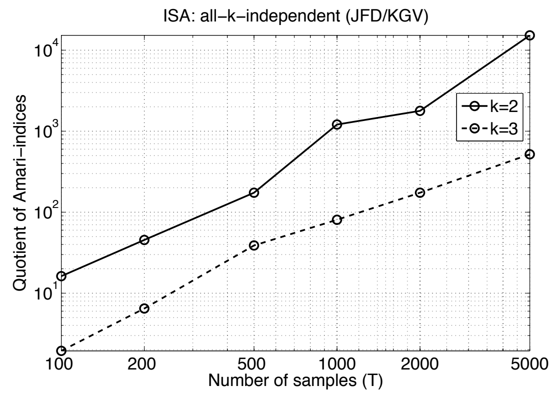
(a)



(b)

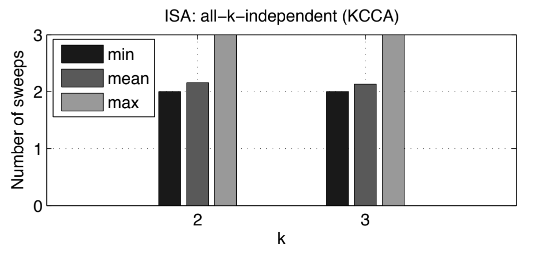


(c)

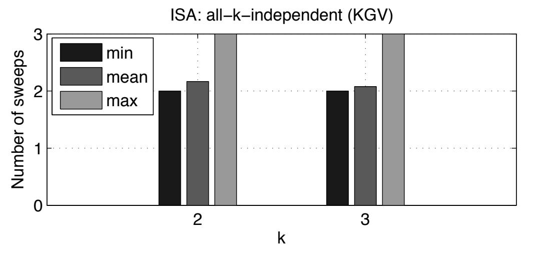


(d)

Figure 12: (a) and (b): Amari-indices of the KCCA and KGV methods, respectively, as a function of the sample number and for $k = 2$ and 3 on the *all-k-independent* database. For more details, see Table 7. (c): ratio of Amari-indices of JFD and KCCA methods, (d): ratio of Amari-indices of JFD and KGV methods.



(a)



(b)

Figure 13: Number of sweeps of permutation optimization for the KCCA (a) and KGV (b) methods for the *all-k-independent* database and for different k values. Black: minimum, gray: mean, light gray: maximum.

	$k=2$	$k=3$
KCCA/KGV	0.0017% (±0.0014)	0.16% (±0.04)

Table 7: The Amari-index of the KCCA and KGV methods for database *all-k-independent* for different k values: average ± deviation. Number of samples: $T = 5,000$. For other sample numbers between $100 \leq T < 5,000$, see Figure 12.

undercomplete version (uBSSD) of the task has been presented, and it has been shown how to derive an independent subspace analysis (ISA) task from the uBSSD problem. Recent developments of the ISA techniques enabled us to handle the emerging high dimensional problems. Our earlier results, namely the ISA Separation Theorem (Szabó et al., 2006b) motivated us to reduce the ISA task to the search for the optimal permutation of the ICA components. The components were grouped with a novel joint decorrelation technique, the joint f-decorrelation (JFD) method (Szabó and Lőrincz, 2006).

Also, we adapted other ICA techniques, such as the KCCA and KGV methods to the ISA task and studied their efficiency. Simulations indicated that although the KCCA and KGV methods give rise to serious computational burden relative to the JFD method, they can be advantageous for smaller ISA tasks and for ISA tasks when the number of samples is small.

Finally, we note that we achieved small errors in these high dimensional computations. These small errors indicate that the Separation Theorem is robust and might be extended to a larger class of noise sources.

Acknowledgments

The authors would like to thank Katalin Lázár and Gábor Szirtes for reading the manuscript, and for their valuable suggestions. We furthermore thank the anonymous reviewers for their helpful comments.

Appendix A. Proof of the ISA Separation Theorem

We shall rely on entropy inequalities (Section A.1). In Section A.2 connection to the ICA cost function is derived (Lemma 2). The ISA Separation Theorem then follows.

A.1 EPI-type Relations

First, consider the so-called entropy power inequality (EPI)

$$e^{2H\left(\sum_{i=1}^{L} u_i\right)} \geq \sum_{i=1}^{L} e^{2H(u_i)},$$

where $u_1, \ldots, u_L \in \mathbb{R}$ denote continuous random variables. This inequality holds, for example, for independent continuous variables (Cover and Thomas, 1991).

If EPI is satisfied on the surface of the L-dimensional unit sphere $\mathbb{S}^L$, then a further inequality holds:

Lemma 1 *Suppose that continuous random variables* $u_1,\dots,u_L \in \mathbb{R}$ *satisfy the following inequality*

$$e^{2H\left(\sum_{i=1}^{L} w_i u_i\right)} \geq \sum_{i=1}^{L} e^{2H(w_i u_i)}, \forall \mathbf{w} \in \mathcal{S}^L. \tag{16}$$

This inequality will be called the w-EPI *condition. Then Equation* (8) *holds, too.*

Proof *Assume that* $\mathbf{w} \in \mathcal{S}^L$. *Applying* ln *on condition* (16), *and using the monotonicity of the* ln *function, we can see that the first inequality is valid in the following inequality chain*

$$2H\left(\sum_{i=1}^{L} w_i u_i\right) \geq \ln\left(\sum_{i=1}^{L} e^{2H(w_i u_i)}\right) = \ln\left(\sum_{i=1}^{L} e^{2H(u_i)} w_i^2\right)$$
$$\geq \sum_{i=1}^{L} w_i^2 \ln\left(e^{2H(u_i)}\right) = 2\sum_{i=1}^{L} w_i^2 H(u_i).$$

Here,

1. *we used the relation*

$$H(w_i u_i) = H(u_i) + \ln(|w_i|)$$

for the entropy of the transformed variable (Cover and Thomas, 1991). Hence

$$e^{2H(w_i u_i)} = e^{2H(u_i)+2\ln(|w_i|)} = e^{2H(u_i)} e^{2\ln(|w_i|)} = e^{2H(u_i)} w_i^2.$$

2. *In the second inequality, the concavity of* ln *was exploited.* □

Note 5 *w-EPI holds, for example, for independent variables* u_i, *because independence is not affected by multiplication with a constant.*

A.2 Connection to the Cost Function of the ICA Task

The ISA Separation Theorem will be a corollary of the following claim:

Lemma 2 *Let* $\mathbf{y} = \left[\mathbf{y}^1;\dots;\mathbf{y}^M\right] = \mathbf{y}(\mathbf{W}) = \mathbf{W}\mathbf{s} \in \mathbb{R}^D$, *where* $\mathbf{W} \in O^D$ $(D = \sum_{m=1}^{M} d_m)$, $\mathbf{y}^m \in \mathbb{R}^{d_m}$ *is the estimation of the* m^{th} *component of the ISA task. Let* $y_i^m \in \mathbb{R}$ *be the* i^{th} *coordinate of the* m^{th} *component* $(i = 1,\dots,d_m)$. *Similarly, let* $s_i^m \in \mathbb{R}$ *stand for the* i^{th} *coordinate of the* m^{th} *source. Let us assume that the* $\mathbf{u} := \mathbf{s}^m \in \mathbb{R}^{d_m}$ *sources satisfy the condition* (8). *Then*

$$\sum_{m=1}^{M}\sum_{i=1}^{d_m} H\left(y_i^m\right) \geq \sum_{m=1}^{M}\sum_{i=1}^{d_m} H\left(s_i^m\right). \tag{17}$$

Proof *Let us denote the* $(i,j)^{th}$ *element of matrix* $\mathbf{W}$ *by* $W_{i,j}$. *Coordinates of* $\mathbf{y}$ *and* $\mathbf{s}$ *will be denoted by* y_i *and* s_i, *respectively. Further, let* $\mathcal{G}^m$ *denote the indices of the* m^{th} *subspace* $(m = 1,\dots,M)$, *that is,* $\mathcal{G}^m := \{1 + \sum_{i=1}^{m-1} d_i, \dots, \sum_{i=1}^{m} d_i\}$ $(d_0 := 0)$. *Now, writing the elements of the* i^{th} *row of matrix multiplication* $\mathbf{y} = \mathbf{W}\mathbf{s}$, *we have*

$$y_i = \sum_{j\in\mathcal{G}^1} W_{i,j} s_j + \ldots + \sum_{j\in\mathcal{G}^M} W_{i,j} s_j \tag{18}$$

and thus,

$$
\begin{aligned}
H(y_i) &= H\left(\sum_{m=1}^{M}\sum_{j\in\mathcal{G}^m} W_{i,j}s_j\right) && (19)\\
&= H\left(\sum_{m=1}^{M}\left[\left(\sum_{l\in\mathcal{G}^m} W_{i,l}^2\right)^{\frac{1}{2}} \frac{\sum_{j\in\mathcal{G}^m} W_{i,j}s_j}{\left(\sum_{l\in\mathcal{G}^m} W_{i,l}^2\right)^{\frac{1}{2}}}\right]\right) && (20)\\
&\geq \sum_{m=1}^{M}\left[\left(\sum_{l\in\mathcal{G}^m} W_{i,l}^2\right) H\left(\frac{\sum_{j\in\mathcal{G}^m} W_{i,j}s_j}{\left(\sum_{l\in\mathcal{G}^m} W_{i,l}^2\right)^{\frac{1}{2}}}\right)\right] && (21)\\
&= \sum_{m=1}^{M}\left[\left(\sum_{l\in\mathcal{G}^m} W_{i,l}^2\right) H\left(\sum_{j\in\mathcal{G}^m}\frac{W_{i,j}}{\left(\sum_{l\in\mathcal{G}^m} W_{i,l}^2\right)^{\frac{1}{2}}} s_j\right)\right] && (22)\\
&\geq \sum_{m=1}^{M}\left[\left(\sum_{l\in\mathcal{G}^m} W_{i,l}^2\right)\sum_{j\in\mathcal{G}^m}\left(\frac{W_{i,j}}{\left(\sum_{l\in\mathcal{G}^m} W_{i,l}^2\right)^{\frac{1}{2}}}\right)^2 H(s_j)\right] && (23)\\
&= \sum_{j\in\mathcal{G}^1} W_{i,j}^2 H(s_j) + \ldots + \sum_{j\in\mathcal{G}^M} W_{i,j}^2 H(s_j). && (24)
\end{aligned}
$$

The above steps can be justified as follows:

1. (19)*: Equation* (18) *was inserted into the argument of H.*
2. (20)*: New terms were added for Lemma 1.*
3. (21)*: Sources $\mathbf{s}^m$ are independent of one another and this independence is preserved upon mixing* within *the subspaces, and we could also use Lemma 1, because $\mathbf{W}$ is an orthogonal matrix.*
4. (22)*: Nominators were transferred into the $\sum_j$ terms.*
5. (23)*: Variables $\mathbf{s}^m$ satisfy condition* (8) *according to our assumptions.*
6. (24)*: We simplified the expression after squaring.*

Using this inequality, summing it for i, exchanging the order of the sums, and making use of the orthogonality of matrix $\mathbf{W}$, we have

$$
\begin{aligned}
\sum_{i=1}^{D} H(y_i) &\geq \sum_{i=1}^{D}\left(\sum_{j\in\mathcal{G}^1} W_{i,j}^2 H(s_j) + \ldots + \sum_{j\in\mathcal{G}^M} W_{i,j}^2 H(s_j)\right)\\
&= \sum_{j\in\mathcal{G}^1}\left(\sum_{i=1}^{D} W_{i,j}^2\right) H(s_j) + \ldots + \sum_{j\in\mathcal{G}^M}\left(\sum_{i=1}^{D} W_{i,j}^2\right) H(s_j)\\
&= \sum_{j=1}^{D} H(s_j). \qquad \square
\end{aligned}
$$

Corollary (ISA Separation Theorem) *ICA minimizes the l.h.s. of Equation* (17)*, that is, it minimizes* $\sum_{m=1}^{M}\sum_{i=1}^{d_m} H\left(y_i^m\right)$*. The set of minima is invariant to permutations and to changes of the signs. Also, according to Proposition 2,* $\{s_i^m\}$*, that is, the coordinates of the* $\mathbf{s}^m$ *components of the ISA task belong to the set of the minima.* □

Appendix B. Kernel Covariance Technique for the ISA Task

For the sake of completeness, the extension of the KC method (Gretton et al., 2005) for the ISA task is detailed below. The extension is similar to the extensions presented in Section 4.1.2, and we use the notations of that section.

First, we would like to measure the dependence of two 2 random variables $\mathbf{u} \in \mathbb{R}^{d_1}$ and $\mathbf{v} \in \mathbb{R}^{d_2}$. The KC technique defines their dependence as their maximal covariance on the unit spheres $\mathcal{S}^{\mathbf{u}}$, $\mathcal{S}^{\mathbf{v}}$ of function spaces $\mathcal{F}^{\mathbf{u}}$, $\mathcal{F}^{\mathbf{v}}$:

$$J_{\mathrm{KC}}(\mathbf{u},\mathbf{v},\mathcal{F}^{\mathbf{u}},\mathcal{F}^{\mathbf{v}}) := \sup_{g\in\mathcal{S}^{\mathbf{u}},h\in\mathcal{S}^{\mathbf{v}}} |E\{[g(\mathbf{u})-Eg(\mathbf{u})][h(\mathbf{v})-Eh(\mathbf{v})]\}|.$$

This function J_{KC} can be estimated empirically from T-element samples $\mathbf{u}_1,\ldots,\mathbf{u}_T \in \mathbb{R}^{d_1}$, $\mathbf{v}_1,\ldots,\mathbf{v}_T \in \mathbb{R}^{d_2}$:

$$J_{\mathrm{KC}}^{emp}(\mathbf{u},\mathbf{v},\mathcal{F}^{\mathbf{u}},\mathcal{F}^{\mathbf{v}}) := \sup_{g\in\mathcal{S}^{\mathbf{u}},h\in\mathcal{S}^{\mathbf{v}}} \left|\frac{1}{T}\sum_{t=1}^{T}[g(\mathbf{u}_t)-\bar{g}][h(\mathbf{v}_t)-\bar{h}]\right|.$$

The estimation can be reduced to the following conditional maximization problem:

$$J_{\mathrm{KC}}^{emp}(\mathbf{u},\mathbf{v},\mathcal{F}^{\mathbf{u}},\mathcal{F}^{\mathbf{v}}) = \sup_{\mathbf{c}_1^*\widetilde{\mathbf{K}}^{\mathbf{u}}\mathbf{c}_1\le 1,\mathbf{c}_2^*\widetilde{\mathbf{K}}^{\mathbf{v}}\mathbf{c}_2\le 1} \mathbf{c}_1^*\widetilde{\mathbf{K}}^{\mathbf{u}}\widetilde{\mathbf{K}}^{\mathbf{v}}\mathbf{c}_2. \tag{25}$$

After the adaptation of the Lagrange multiplier technique and the computation of the stationary points of (25) it can be realized that the values of $[\mathbf{c}_1;\mathbf{c}_2]$ and J_{KC}^{emp} can be computed as the solutions of the generalized eigenvalue problem

$$\begin{pmatrix} \widetilde{\mathbf{K}}^{\mathbf{u}} & \widetilde{\mathbf{K}}^{\mathbf{u}}\widetilde{\mathbf{K}}^{\mathbf{v}} \\ \widetilde{\mathbf{K}}^{\mathbf{v}}\widetilde{\mathbf{K}}^{\mathbf{u}} & \widetilde{\mathbf{K}}^{\mathbf{v}} \end{pmatrix}\begin{pmatrix}\mathbf{c}_1\\ \mathbf{c}_2\end{pmatrix} = \lambda\begin{pmatrix} \widetilde{\mathbf{K}}^{\mathbf{u}} & \mathbf{0} \\ \mathbf{0} & \widetilde{\mathbf{K}}^{\mathbf{v}} \end{pmatrix}\begin{pmatrix}\mathbf{c}_1\\ \mathbf{c}_2\end{pmatrix}. \tag{26}$$

If the task is to measure the dependence between more than two random variables $\mathbf{y}^1 \in \mathbb{R}^{d_1},\ldots,$ $\mathbf{y}^M \in \mathbb{R}^{d_M}$ then (26) is to be replaced with the following generalized eigenvalue problem:

$$\begin{pmatrix} \widetilde{\mathbf{K}}^1 & \widetilde{\mathbf{K}}^1\widetilde{\mathbf{K}}^2 & \cdots & \widetilde{\mathbf{K}}^1\widetilde{\mathbf{K}}^M \\ \widetilde{\mathbf{K}}^2\widetilde{\mathbf{K}}^1 & \widetilde{\mathbf{K}}^2 & \cdots & \widetilde{\mathbf{K}}^2\widetilde{\mathbf{K}}^M \\ \vdots & \vdots & & \vdots \\ \widetilde{\mathbf{K}}^M\widetilde{\mathbf{K}}^1 & \widetilde{\mathbf{K}}^M\widetilde{\mathbf{K}}^2 & \cdots & \widetilde{\mathbf{K}}^M \end{pmatrix}\begin{pmatrix}\mathbf{c}_1\\ \mathbf{c}_2\\ \vdots\\ \mathbf{c}_M\end{pmatrix} = \lambda\begin{pmatrix} \widetilde{\mathbf{K}}^1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \widetilde{\mathbf{K}}^2 & \cdots & \mathbf{0} \\ \vdots & \vdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \widetilde{\mathbf{K}}^M \end{pmatrix}\begin{pmatrix}\mathbf{c}_1\\ \mathbf{c}_2\\ \vdots\\ \mathbf{c}_M\end{pmatrix}.$$

Using the maximal eigenvalue of this problem, J_{KC} can be estimated.

References

Karim Abed-Meraim and Adel Belouchrani. Algorithms for joint block diagonalization. In *Proceedings of European Signal Processing Conference (EUSIPCO 2004)*, pages 209–212, 2004.

Shotaro Akaho, Yasuhiko Kiuchi, and Shinji Umeyama. MICA: Multimodal independent component analysis. In *Proceedings of International Joint Conference on Neural Networks (IJCNN '99)*, pages 927–932, 1999.

Ian F. Akyildiz, WellJan Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, 2002.

Shun-ichi Amari, Andrzej Cichocki, and Howard H. Yang. A new learning algorithm for blind signal separation. *Advances in Neural Information Processing Systems*, 8:757–763, 1996.

Shoko Araki, Shoji Makino, Ryo Mukai, Tsuyoki Nishikawa, and Hiroshi Saruwatari. Fundamental limitation of frequency domain blind source separation for convolved mixture of speech. *IEEE Transactions on Speech and Audio Processing*, 11(2):109–116, 2003.

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

Francis R. Bach and Michael I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.

Francis R. Bach and Michael I. Jordan. Beyond independent components: Trees and clusters. *Journal of Machine Learning Research*, 4:1205–1233, 2003.

Jean-François Cardoso. Multidimensional independent component analysis. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP '98)*, volume 4, pages 1941–1944, Seattle, WA, USA, 1998.

Andrzej Cichocki and Shun-ichi Amari. *Adaptive Blind Signal and Image Processing*. John Wiley & Sons, 2002.

Pierre Comon. Independent component analysis, a new concept? *Signal Processing*, 36:287–314, 1994.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley and Sons, New York, USA, 1991.

Fani Deligianni, Benny Lo, and Guang-Zhong Yang. Source recovery for body sensor network. In *Proceedings of International Workshop on Wearable and Implantable Body Sensor Networks 2006 (BSN 2006)*, pages 199–202, 2006.

Scott C. Douglas, Hiroshi Sawada, and Shoji Makino. Natural gradient multichannel blind deconvolution and speech separation using causal FIR filters. *IEEE Transactions on Speech and Audio Processing*, 13(1):92–104, 2005.

Mads Dyrholm, Scott Makeig, and Lars Kai Hansen. Model selection for convolutive ICA with an application to spatio-temporal analysis of EEG. *Neural Computation*, apr 2007.

Alan Edelman, Tomas Arias, and Steven T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, 1998.

Kai-Tai Fang, Samuel Kotz, and Kai Wang Ng. *Symmetric Multivariate and Related Distributions*. Chapman and Hall, 1990.

Cédric Févotte and Christian Doncarli. A unified presentation of blind source separation for convolutive mixtures using block-diagonalization. In *Proceedings of Independent Component Analysis and Blind Signal Separation (ICA 2003)*, pages 349–354, Nara, Japan, 2003.

Kenji Fukumizu, Francis R. Bach, and Arthur Gretton. Statistical consistency of kernel canonical correlation analysis. *Journal of Machine Learning Research*, 8:361–383, 2007.

Gary H. Glover. Deconvolution of impulse response in event-related BOLD fMRI. *NeuroImage*, 9: 416–429, 1999.

Arthur Gretton, Alexander Smola, Olivier Bousquet, and Bernhard Schölkopf. Kernel methods for measuring independence. *Journal of Machine Learning Research*, 6:2075–2129, 2005.

John B. Hedgepeth, Vincent F. Gallucci, F. O'Sullivan, and Richard E. Thorne. An expectation maximization and smoothing approach for indirect acoustic estimation of fish size and density. *ICES Journal of Marine Science*, 56(1):36–50, 1999.

Aapo Hyvärinen and Patrik O. Hoyer. Emergence of phase and shift invariant features by decomposition of natural images into independent feature subspaces. *Neural Computation*, 12:1705–1720, 2000.

Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent Component Analysis*. John Wiley & Sons, 2001.

Aapo Hyvärinen and Urs Köster. FastISA: A fast fixed-point algorithm for independent subspace analysis. In *Proceedings of European Symposium on Artificial Neural Networks (ESANN 2006)*, Bruges, Belgium, 2006.

Aapo Hyvärinen and Erkki Oja. A fast fixed-point algorithm for independent component analysis. *Neural Computation*, 9(7):1483–1492, 1997.

Tzyy-Ping Jung, Scott Makeig, Te-Won Lee, Martin J. McKeown, Glen Brown, Anthony J. Bell, and Terrence J. Sejnowski. Independent component analysis of biomedical signals. In *Proceedings of International Workshop on Independent Component Analysis and Signal Separation (ICA 2000)*, pages 633–644, Helsinki, 2000.

Christian Jutten and Jeanny Herault. Blind separation of sources: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, 24:1–10, 1991.

Hakan Karslı. Further improvement of temporal resolution of seismic data by autoregressive (AR) spectral extrapolation. *Journal of Applied Geophysics*, 59:324–336, 2006.

Tuvia Kotzer, Nir Cohen, and Joseph Shamir. Generalized projection algorithms with applications to optics and signal restoration. *Optics Communications*, 156(1):77–91, 1998.

Ross A. Lippert. *Nonlinear Eigenvalue Problems*. PhD thesis, Massachusetts Institute of Technology, 1998.

Adam MacDonald and Stephen Cain. Derivation and application of an anisoplanatic optical transfer function for blind deconvolution of laser radar imagery. *Unconventional Imaging*, 5896:9–20, 2005.

Nikolaos Mitianoudis and Michael E. Davies. Audio source separation of convolutive mixtures. *IEEE Transactions on Speech and Audio Processing*, 11(5):489–497, 2003.

Yasunori Nishimori, Shotaro Akaho, and Mark D. Plumbley. Riemannian optimization method on the flag manifold for independent subspace analysis. In *Proceedings of Independent Component Analysis and Blind Signal Separation (ICA 2006)*, volume 3889 of *LNCS*, pages 295–302. Springer, 2006.

Guido Nolte, Frank C. Meinecke, Andreas Ziehe, and Klaus-Robert Müller. Identifying interactions in mixed and noisy complex systems. *Physical Review E*, 73(051913), 2006.

Michael S. Pedersen, Jan Larsen, Ulrik Kjems, and Lucas C. Parra. A survey of convolutive blind source separation methods. In *Springer Handbook of Speech (to appear)*. Springer Press, sep 2007. URL `http://www2.imm.dtu.dk/pubdb/p.php?4924`.

Mark D. Plumbley. Lie group methods for optimization with orthogonality constraints. In *Proceedings of Independent Component Analysis and Blind Signal Separation (ICA 2004)*, volume 3195 of *LNCS*, pages 1245–1252. Springer, 2004.

Barnabás Póczos and András Lőrincz. Independent subspace analysis using geodesic spanning trees. In *Proceedings of International Conference on Machine Learning (ICML 2005)*, pages 673–680, Bonn, Germany, 2005a.

Barnabás Póczos and András Lőrincz. Independent subspace analysis using k-nearest neighborhood distances. *Artificial Neural Networks: Formal Models and their Applications - ICANN 2005, pt 2, Proceedings*, 3697:163–168, 2005b.

Barnabás Póczos and András Lőrincz. Non-combinatorial estimation of independent autoregressive sources. *Neurocomputing Letters*, 69:2416–2419, 2006.

Nicolas Quinquis, Isao Yamada, and Kohichi Sakaniwa. Efficient dual Cayley parametrization technique for ICA with orthogonality constraints. In *Proceedings of ICA Research Network International Workshop (ICARN 2006)*, pages 123–126, Liverpool, U.K., 2006.

Ravikiran Rajagopal and Lee C. Potter. Multivariate MIMO FIR inverses. *IEEE Transactions on Image Processing*, 12:458 – 465, 2003.

Michael J. Roan, Mark R. Gramann, Josh G. Erling, and Leon H. Sibul. Blind deconvolution applied to acoustical systems identification with supporting experimental results. *The Journal of the Acoustical Society of America*, 114(4):1988–1996, 2003.

Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola. *Advances in Kernel Methods - Support Vector Learning*. MIT Press, Cambridge, MA, 1999.

Harald Stögbauer, Alexander Kraskov, Sergey A. Astakhov, and Peter Grassberger. Least dependent component analysis based on mutual information. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 70(066123), 2004.

Zoltán Szabó and András Lőrincz. Real and complex independent subspace analysis by generalized variance. In *Proceedings of ICA Research Network International Workshop (ICARN 2006)*, pages 85–88, Liverpool, U.K., 2006. http://arxiv.org/abs/math.ST/0610438.

Zoltán Szabó, Barnabás Póczos, and András Lőrincz. Cross-entropy optimization for independent process analysis. In *Proceedings of Independent Component Analysis and Blind Signal Separation (ICA 2006)*, volume 3889 of *LNCS*, pages 909–916. Springer, 2006a.

Zoltán Szabó, Barnabás Póczos, and András Lőrincz. Separation theorem for $\mathbb{K}$-independent subspace analysis with sufficient conditions. Technical report, Eötvös Loránd University, Budapest, 2006b. http://arxiv.org/abs/math.ST/0608100.

Seiji Takano. The inequalities of Fisher information and entropy power for dependent variables. In *Symposium on Probability Theory and Mathematical Statistics*, 1995.

Fabian J. Theis. Uniqueness of complex and multidimensional independent component analysis. *Signal Processing*, 84(5):951–956, 2004.

Fabian J. Theis. Blind signal separation into groups of dependent signals using joint block diagonalization. In *Proceedings of International Society for Computer Aided Surgery (ISCAS 2005)*, pages 5878–5881, Kobe, Japan, 2005a.

Fabian J. Theis. Multidimensional independent component analysis using characteristic functions. In *Proceedings of European Signal Processing Conference (EUSIPCO 2005)*, 2005b.

Fabian J. Theis. Towards a general independent subspace analysis. In *Proceedings of Neural Information Processing Systems (NIPS 2006)*, 2006.

Roland Vollgraf and Klaus Obermayer. Multi-dimensional ICA to separate correlated sources. In *Proceedings of Neural Information Processing Systems (NIPS 2001)*, volume 14, pages 993–1000, 2001.

Cabir Vural and William A. Sethares. Blind image deconvolution via dispersion minimization. *Digital Signal Processing*, 16:137–148, 2006.

Grace Wahba. Support vector machines, reproducing kernel hilbert spaces, and randomized GACV. In *Advances in Kernel Methods*, pages 69–88. MIT Press, 1999.

Journal of Machine Learning Research 8 (2007) 1097-1111 Submitted 9/06; Revised 2/07; Published 5/07

Bilinear Discriminant Component Analysis

Mads Dyrholm DYRHOLM@ENGR.CCNY.CUNY.EDU
Christoforos Christoforou CCHRISTOFOROU@GC.CUNY.EDU
Lucas C. Parra PARRA@CCNY.CUNY.EDU
The City College of The City University of New York
Convent Avenue @ 138th Street
New York, NY 10031, USA

Editor: Leslie Pack Kaelbling

Abstract

Factor analysis and discriminant analysis are often used as complementary approaches to identify linear components in two dimensional data arrays. For three dimensional arrays, which may organize data in dimensions such as space, time, and trials, the opportunity arises to combine these two approaches. A new method, Bilinear Discriminant Component Analysis (BDCA), is derived and demonstrated in the context of functional brain imaging data for which it seems ideally suited. The work suggests to identify a subspace projection which optimally separates classes while ensuring that each dimension in this space captures an independent contribution to the discrimination.

Keywords: bilinear, decomposition, component, classification, regularization

1. Introduction

The work presented in this paper is motivated by the analysis of functional brain imaging signals recorded with functional magnetic resonance imaging (fMRI) or electric or magnetic encephalography (EEG/MEG). These imaging modalities record brain activity across time at multiple locations, providing spatio-temporal data. The design of a brain imaging experiment often includes multiple repetitions or trials. Trials may differ in the type of stimulus presented, the task given to the subject, or the subject's response. Hence, brain imaging data is often given as a three-dimensional array including space, time, and trials. In addition, for each trial we have labels at our disposal that characterize the trial.

A number of linear analysis methods have been proposed for both EEG/MEG and fMRI in order to decompose this data into meaningful linear 'components'. These methods include principal component analysis (Squires et al., 1977; Bullmore et al., 1996), independent component analysis (Makeig et al., 1996; Calhoun et al., 2001), and linear discriminant analysis (Mørch et al., 1997; Parra et al., 2005), denoted respectively as PCA, ICA, and LDA. Linear decomposition of a data matrix $\mathbf{X} \in \mathbb{R}^{D\times T}$ using PCA or ICA involves estimation of the factors of the model

$$(\mathbf{X})_{ij} \approx \sum_{k=1}^{K} (\mathbf{a}_k)_i (\mathbf{s}_k)_j$$

where $\mathbf{a}_k \in \mathbb{R}^D$, $\mathbf{s}_k \in \mathbb{R}^T$, and K denotes the number of components in the model. Uniqueness of such decomposition is guaranteed by declaring additional conditions on the factors $\{\mathbf{a}_k\}$ and $\{\mathbf{s}_k\}$, that is, assuming orthogonality in the case of PCA or assuming independence in the case of ICA.

When applying PCA or ICA to brain imaging data, trials are often combined with time samples to form a single dimension, thereby ignoring the tensor structure of the data (see, e.g., Delorme and Makeig, 2004).

A related model which aims to exploit the additional structure in the data provided by the third dimension is parallel factor analysis (PARAFAC) (Harshman, 1970). In PARAFAC the three-way data array $\mathbf{X} \in \mathbb{R}^{D\times T\times N}$ is decomposed under the model

$$(\mathbf{X}_n)_{ij} \approx \sum_{k=1}^{K} (\mathbf{a}_k)_i (\mathbf{b}_k)_j (\mathbf{c}_k)_n \tag{1}$$

where $\mathbf{a}_k \in \mathbb{R}^D$, $\mathbf{b}_k \in \mathbb{R}^T$, $\mathbf{c}_k \in \mathbb{R}^N$, and $\cdot \approx \cdot$ denotes least-squares approximation. This model has been suggested as a tool to analyze for instance EEG (Harshman, 1970; Möcks, 1988; Miwakeichi et al., 2004; Martinez-Montes et al., 2004; Mørup et al., 2006) and fMRI (Andersen and Rayens, 2004; Beckmann and Smith, 2005). The PARAFAC decomposition often turns out to be unique even without having to declare any additional assumptions such as orthogonality or independence among the factors of the model (Kruskal, 1977).[1] Each component in the PARAFAC model, say component k consisting of $\{\mathbf{a}_k, \mathbf{b}_k, \mathbf{c}_k\}$, offers a simple interpretation—for example, when having data on the form of spatio-temporal matrices recorded in repeated trials: $\mathbf{a}_k$ will describe the spatial distribution of the temporal source signal $\mathbf{b}_k$ with relative strengths in the different trials given by the elements of $\mathbf{c}_k$. In this example index i labels the space coordinate, index j refers to the time coordinate, and index n enumerates the trials.

A limitation of the unsupervised methods PARAFAC, ICA and PCA, is that the available labels are not used when identifying components in the data. Typically, the variability in the data due to noise and task irrelevant activity is quite large when compared to the signal that relates specifically to the experimental question under consideration. In fMRI data, for instance, the activity that is extracted from the raw data is often only a small fraction of the total background BOLD signal. A similar situation arises in EEG where often hundreds of trials have to be averaged to gain a significant difference between two experimental conditions. Linear discrimination methods have been suggested in both EEG and fMRI to compensate for this problem (Mørch et al., 1997; Parra et al., 2005). In essence, these methods project the data onto a linear subspace that best characterizes the relevant activity. This stands in contrast to an unsupervised analysis that decomposes the data into orientations that capture the largest variability in the data. Hence, unsupervised methods that are purely variance based (PCA and PARAFAC) may fail to recover components with very small signal-to-noise ratios (SNR), typically less than -20dB for EEG. Unsupervised ICA can in principle recover components with low SNR (see, e.g., Beckmann and Smith, 2005), if the ICA decomposition is followed by component inspection in order to identify task specific components. However, dimensionality reduction (typically PCA) may remove task specific components with low SNR which will then no longer be recovered with subsequent ICA. In this paper we propose an algorithm that includes the labels at the earliest stage in order to identify possible subspaces wherein the SNR of task specific activity is maximized.

We propose to find a subspace projection in which the dimensions sum up to an optimal classification of the trials, while each dimension contributes to this discriminant sum independently across trials. The subspace will be restricted to a bilinear subspace to express the assumption that each contributing dimensions should have a fixed spatial profile and an associated temporal profile. The

1. Here ‘unique’ means: unique except for trivial scaling and permutation ambiguities.

underlying task relevant activity can be expected to involve a number of interacting sources, and we therefore allow the bilinear subspace to be of rank-K where $K > 1$. All this can be compactly represented in a factorization of the form

$$(\mathbf{X}_n^{(\mathcal{W})})_{ij} = \sum_{k=1}^{K} (\mathbf{a}_k)_i (\mathbf{b}_k)_j (\mathbf{c}_k)_n \tag{2}$$

where $\mathbf{X}_n^{(\mathcal{W})}$ denotes projected data in discriminant directions $\{\mathbf{a}_k\}$ and $\{\mathbf{b}_k\}$, with statistically independent $(\mathbf{c}_k)_n$ across n.

As a first step, a manifold of possible $\{\mathbf{a}_k, \mathbf{b}_k\}$ is identified using Bilinear Discriminant Analysis (BLDA), extending the work of Dyrholm and Parra (2006). Any choice within this manifold gives the same (optimal) classification. To select a specific $\{\mathbf{a}_k, \mathbf{b}_k\}$ we require that the resulting $\{\mathbf{c}_k\}$ are independent across n. The resulting model can potentially identify relevant components in low SNR by using the information available in the labels.

The model satisfies a factorization as in PARAFAC, but the parameter estimation satisfies different optimality criteria, namely discriminability and independence. Hence, we are proposing a factorized model where the components are independently discriminant.

ICA in tensor models has been suggested before by, for example, Beckmann and Smith (2005); unsupervised dimensionality reduction in tensorial data is reviewed by De Lathauwer and Vandewalle (2004); supervised learning in the context of ICA has been investigated by, for example, Sakaguchi et al. (2002); however, our method uniquely enables supervised dimensionality reduction and learning of ICA in a tensor model.

2. Bilinear Discriminant Analysis

The aim of Linear Discriminant Analysis (LDA) is to find a set of weights $\mathbf{w}$ and a threshold ε such that the discriminant function

$$t(\mathbf{x}_n) = \mathbf{w}^{\mathrm{T}}\mathbf{x}_n - \varepsilon \tag{3}$$

maximizes a discrimination criterion, for example, in a two class problem, the data vector $\mathbf{x}_n$ is assigned to one class if $t(\mathbf{x}_n) > 0$ and to the other class if $t(\mathbf{x}_n) < 0$. Methods for determining the weights $\mathbf{w}$, and the threshold ε, include Least-Squares Regression, Logistic Regression, Fisher's Linear Discriminant and the Single-Layer Perceptron (see, e.g., Bishop, 1996; McCulloch and Searle, 2001). The simplicity of the LDA model (as opposed to more complex non-linear models) makes it a good candidate for classification in situations where training data is very limited (see, e.g., Mørch et al., 1997), which is typically the case, for instance, in Brain Computer Interfacing (BCI). Furthermore, LDA allows identification of class dependent features/components in the data (Parra et al., 2005; Ye, 2005).

In this paper we address situations where data is available as a set of matrices $\{\mathbf{X}_n\}$ instead of as a set of vectors $\{\mathbf{x}_n\}$. For example, in brain imaging modalities (e.g., fMRI or EEG) the columns of $\mathbf{X}_n$ can represent space while its rows represent time. LDA is directly applicable in the form (3) by letting the data vector $\mathbf{x}_n$ be a stacking of the elements of the data matrix $\mathbf{X}_n$, but, the data matrix structure could potentially be exploited to obtain a more parsimonious representation of the weight vector $\mathbf{w}$. We consider situations where the data is not only given as a set of matrices, but also, the generators of class-dependent variance in the data have low-rank contributions to each data matrix $\mathbf{X}_n$. In EEG for instance, an electrical current source which is spatially static in the brain will give a rank-one contribution to the spatiotemporal $\mathbf{X}_n$ (see also Makeig et al., 1996; Dyrholm and Parra, 2006). In this paper we incorporate a low-rank assumption in LDA by generalizing (3) to the following form

$$t(\mathbf{X}_n) = \mathrm{Trace}(\mathbf{U}^{\mathrm{T}}\mathbf{X}_n\mathbf{V}) - \varepsilon \tag{4}$$

where $\mathbf{U}$ and $\mathbf{V}$ are parameter matrices which share the same number of columns. Let R denote the number of columns in both $\mathbf{U}$ and $\mathbf{V}$, then (4) is equivalent to (3) but with a rank-R constraint on $\mathbf{w}$, that is,

$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_n = \sum_{i,j}(\mathbf{W})_{ij}(\mathbf{X}_n)_{ij} \quad \text{where} \quad \mathbf{W} = \mathbf{U}\mathbf{V}^{\mathrm{T}}. \tag{5}$$

Since $\mathbf{X}_n$ is $D \times T$-dimensional, the number of parameters in the rank-R model (4) is $[R \times (D+T)+1]$ while the number of parameters in the unconstrained LDA model (3) is $[D \times T + 1]$. Thus, if R is kept moderately small compared to $\min\{D,T\}$, improved generalization performance can be expected in limited data, particularly in data where the low-rank generator assumption is fundamentally valid. In this model the parameter R quantifies the number of generators (or current sources for EEG).

Bilinear parameter space factorization has been suggested before in context of Fisher LDA (Visani et al., 2005). In this paper we generalize maximum likelihood Logistic Regression to the case of a bilinear factorization of $\mathbf{w}$. The algorithm can be found in Appendix A. The proposed algorithm has the advantage that it allows us to regularize the estimation and incorporate prior assumptions about smoothness as described in Section 2.1. In Section 4.1 we test the performance of this classifier in a benchmark data set from the BCI Competition 2003, and in Section 4.2 we extract components from the EEG of six human subjects in a rapid serial visual presentation paradigm.

2.1 Smoothness Regularization using Gaussian Processes

For a factorized weight representation regularization can be applied in the column space of $\mathbf{X}_n$ and in the row space separately. For instance, if knowledge is available about the smoothness in the column space of $\mathbf{X}_n$ (e.g., spatial smoothness) or in its row space (e.g., temporal smoothness), such knowledge can be incorporated by declaring a prior p.d.f. on the columns of $\mathbf{U}$ or $\mathbf{V}$ respectively (Dyrholm and Parra, 2006). Spatial and temporal smoothness is typically a valid assumption in EEG and fMRI, see, for example, Penny et al. (2005).

One way to incorporate prior assumptions in the estimation is through the posterior distribution of the weights. Here, we propose to estimate $\mathbf{U}$ and $\mathbf{V}$ through maximization of the posterior. Let $\mathbf{u}_k$ denote the kth column of $\mathbf{U}$, and let $\mathbf{v}_k$ denote the kth column of $\mathbf{V}$. We declare Gaussian Process priors for $\mathbf{u}_k$ and $\mathbf{v}_k$, that is, assume $\mathbf{u}_k \sim \mathcal{N}(\mathbf{0},\mathbf{K}_u)$ and $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0},\mathbf{K}_v)$, where the covariance matrices $\mathbf{K}_u$ and $\mathbf{K}_v$ define the degree and form of smoothness of $\mathbf{u}_k$ and $\mathbf{v}_k$ respectively. This is done through choice of covariance function: Let r be a spatial or temporal measure in context of $\mathbf{X}_n$. For instance r is a measure of spatial distance between data acquisition sensors, or a measure of time difference between two samples in the data. Then a covariance function $\mathrm{k}(r)$ expresses the degree of correlation between any two points with that given distance. For example, a class of covariance functions that has been suggested for modelling smoothness in physical processes, the Matérn class, is given by

$$\mathrm{k}_{\text{Matérn}}(r) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}r}{l}\right)^{\nu}\mathrm{B}\left(\frac{\sqrt{2\nu}r}{l}\right) \tag{6}$$

where l is a length-scale parameter, and ν is a shape parameter. The parameter l can roughly be thought of as the distance within which points are significantly correlated (Rasmussen and Williams, 2006). The parameter ν defines the degree of ripple. The covariance matrix $\mathbf{K}$ is then built by evaluating the covariance function

$$(\mathbf{K})_{ij} = \sigma^2\, \mathrm{k}_{\text{Matérn}}(r_{ij})$$

where r_{ij} is exemplified by the distance between sensor-i and sensor-j, or time difference between sample-i and sample-j, and σ^2 defines the overall parameter scale. Note that there is a scaling ambiguity between $\mathbf{u}$ and $\mathbf{v}$, and the variance parameter σ^2 can thus be kept equal for $\mathbf{u}$ and $\mathbf{v}$. $\mathrm{B}(\cdot)$ is a modified Bessel function (Rasmussen and Williams, 2006). The equations for regularized (maximum posterior) estimation of $\mathbf{U}$ and $\mathbf{V}$ are deferred to Appendix A.1.

3. Subspace Factorization with Labeled Mode Independence

Applying the Bilinear Discriminant Analysis algorithm of the appendix to classify matrices $\mathbf{X}_n$ will deliver estimates $\hat{\mathbf{U}}$ and $\hat{\mathbf{V}}$. Note, however, that these estimates will be subject to an arbitrary linear column space transformation, since

$$\begin{aligned}\text{Trace}(\hat{\mathbf{U}}^{\mathrm{T}}\mathbf{X}_n\hat{\mathbf{V}}) &= \text{Trace}(\mathbf{G}^{-1}\mathbf{G}\hat{\mathbf{U}}^{\mathrm{T}}\mathbf{X}_n\hat{\mathbf{V}}) \\ &= \text{Trace}((\hat{\mathbf{U}}\mathbf{G}^{\mathrm{T}})^{\mathrm{T}}\mathbf{X}_n(\hat{\mathbf{V}}\mathbf{G}^{-1})) \\ &= \text{Trace}(\tilde{\mathbf{U}}^{\mathrm{T}}\mathbf{X}_n\tilde{\mathbf{V}})\end{aligned} \tag{7}$$

where $\mathbf{G} \in \mathbb{R}^{R\times R}$ is arbitrary with full rank, and we have implicitly defined $\tilde{\mathbf{U}} \equiv \hat{\mathbf{U}}\mathbf{G}^{\mathrm{T}}$ and $\tilde{\mathbf{V}} \equiv \hat{\mathbf{V}}\mathbf{G}^{-1}$. Due to this ambiguity, the columns of $\hat{\mathbf{U}}$ will be arbitrarily mixed, and the columns of $\hat{\mathbf{V}}$ will be mixed correspondingly (transposed inverse mix). We propose to enhance the task relevant activity by projecting the data using $\{\tilde{\mathbf{U}}, \tilde{\mathbf{V}}\}$ (similar to the argument of the Trace operator in Equation 7), hence we need a meaningful way to estimate $\mathbf{G}$. As we will show, the projection can be defined along with a criterion for estimating $\mathbf{G}$ such that the entities $\tilde{\mathbf{U}}$ and $\tilde{\mathbf{V}}$ become rather meaningful.

First, we define the projection that uses $\tilde{\mathbf{U}}$ and $\tilde{\mathbf{V}}$, that is, assuming $\mathbf{G}$ is given. Define $\tilde{\mathbf{W}}_r$ as the outer product of the rth columns of $\tilde{\mathbf{U}}$ and $\tilde{\mathbf{V}}$. Least-squares projection of a data matrix $\mathbf{X}_n$ onto the matrix space $\mathcal{W} = \text{span}\{\tilde{\mathbf{W}}_r\}$ is given by

$$\tilde{\mathbf{X}}_n^{(\mathcal{W})} = \text{vec}^{\text{-1}}\left[(\tilde{\mathbf{V}}\,|\!\otimes\!|\,\tilde{\mathbf{U}})(\tilde{\mathbf{V}}\,|\!\otimes\!|\,\tilde{\mathbf{U}})^{+}\,\text{vec}(\mathbf{X}_n)\right] \tag{8}$$

where $|\!\otimes\!|$ denotes the *columnwise* Kronecker product which is also known as the Khatri-Rao product (Bro, 1998), $(\cdot)^{+}$ denotes Moore-Penrose (least-squares pseudo) inverse, and $\text{vec}(\cdot)$ produces a vector by stacking the columns of its matrix argument (and $\text{vec}^{\text{-1}}$ does the opposite). That is, each data trial is projected to the best (in terms of squared residuals) rank-R representation available through the basis $\{\tilde{\mathbf{W}}_r\}$ (the basis is of size R and each $\tilde{\mathbf{W}}_r$ is rank-one).

Next, we define a criterion for estimating $\mathbf{G}$. We design the solution so that the dimensions of the projection act as independently as possible across trials. Our reasoning behind this choice is that different cortical processes might respond to the same stimuli, and are hence not temporally independent, but the trial-to-trial variability between the different cortical networks might satisfy the independence criterion better. That is, by making the component activations as independent as possible across trials, we hope to be able to segregate activity arising from different cortical networks into separate sets of components. The projection (8) can be rewritten in terms of a precise definition of 'component activations' $\{\hat{\mathbf{s}}_n\}$,

$$\tilde{\mathbf{X}}_n^{(\mathcal{W})} = \text{vec}^{-1}\left[\tilde{\mathbf{A}}\tilde{\mathbf{s}}_n\right] \quad \text{where} \quad \tilde{\mathbf{A}} \equiv (\hat{\mathbf{V}}\mathbf{G}^{-1}|\otimes|\,\hat{\mathbf{U}}\mathbf{G}^{\mathrm{T}}) \quad \text{and} \quad \tilde{\mathbf{s}}_n \equiv \tilde{\mathbf{A}}^{+}\,\text{vec}(\mathbf{X}_n)$$

where the $R \times N$ elements of $\{\tilde{\mathbf{s}}_n\}$ are assumed i.i.d. Hence, determining $\mathbf{G}$ is essentially an ICA problem where the 'mixing' matrix $\mathbf{A}$ is parameterized in terms of the elements of $\mathbf{G}$ (given $\hat{\mathbf{U}}$ and $\hat{\mathbf{V}}$). The algorithm for gradient based maximum likelihood estimation of $\mathbf{G}$ is deferred to Appendix B. With an estimate of $\mathbf{G}$, and hence unambiguous estimates of $\tilde{\mathbf{U}} \equiv \hat{\mathbf{U}}\mathbf{G}^{\mathrm{T}}$ and $\tilde{\mathbf{V}} \equiv \hat{\mathbf{V}}\mathbf{G}^{-1}$, the projection (8) can be written

$$(\tilde{\mathbf{X}}_n^{(\mathcal{W})})_{ij} = \sum_{r=1}^{R} (\tilde{\mathbf{U}})_{ir}(\tilde{\mathbf{V}})_{jr}(\tilde{\mathbf{s}}_n)_r \tag{9}$$

The structure of (9) is identical to that of PARAFAC (1), with $R = K$, $\mathbf{a}_k$ being the kth column of $\tilde{\mathbf{U}}$, $\mathbf{b}_k$ being the kth column of $\tilde{\mathbf{V}}$, and $(\mathbf{c}_k)_n$ being $(\tilde{\mathbf{s}}_n)_k$, hence we have derived (2) which we will refer to as a Bilinear Discriminant Component Analysis (BDCA) model.

4. Experiments

We conducted two experiments on real EEG data. The first experiment benchmarks the classification performance of our BLDA method with smoothness regularization against state-of-the art methods in a published data set. In the second experiment we extract meaningful components, using BDCA, and illustrate the usefulness of the method in terms of interpretability.

4.1 Experiment I: Classification Benchmark with 'The BCI Competition 2003'

The EEG data set in this experiment was made available though 'The BCI Competition 2003' (Blankertz et al., 2002, 2004, Data Set IV). The 28 channel EEG was recorded from a single subject performing a 'self-paced key typing', that is, pressing with the index and little fingers corresponding keys in a self-chosen order and timing. Typing was done at an average speed of 1 key per second. Trial matrices were extracted by epoching the data starting 630ms before each key-press. A total of 416 epochs were recorded, each of length 500ms. For the competition, the first 316 epochs were to be used for classifier training, while the remaining 100 epochs were to be used as a test set. Data were recorded at 1000 Hz with a pass-band between 0.05 and 200 Hz, then downsampled to 100Hz sampling rate.

4.1.1 TRAINING AND RESULTS

We tuned the smoothness prior parameters using cross validation on the training set using only a single component, that is, R=1. We used the Matérn class of covariance functions for incorporating smoothness regularization in the model c.f. Section 2.1. Temporal smoothness was implemented by letting r_{ij} equal the normalized temporal latency $|i-j|$ between samples i and j. The relative 3D electrode coordinates were looked up from a standard table provided by Delorme and Makeig (2004), and spatial smoothness was implemented by letting r_{ij} equal the Euclidean distance between electrodes i and j in a normalized space where the human head was assumed spherical with radius 0.5. The parameters of the Gaussian Process priors were picked to maximize the area under the ROC

	Std.dev. σ		Matérn scale l		Matérn shape ν		Cross valid.
Experiment	space	time	space	time	space	time	AUC
BCI Competition 2003	0.5	0.5	0.1	15	100	2.5	0.91
RSVP	0.1	0.1	0.1	10	100	2.5	0.95

Table 1: The parameter values for the Matérn covariance function (6) are summarized in this table. The values were found by optimizing AUC using five-fold cross validation. Note that the scale parameter l is defined spatially on a head with unitless radius of 0.5, and defined temporally in terms of samples ($l = 15$ corresponds to 150ms for the BCI Competition data, while $l = 10$ corresponds to 78ms for the RSVP data).

curve ('AUC', see Fawcett, 2003) using five-fold cross validation. The resulting set of parameter values are summarized in the first row of Table 1, and the resulting number of components was $R = 1$.

Benchmark performance was measured on the test set which had not been used during either training or cross validation. The number of misclassified trials in the test set was 21 which places our method on a new third place given the result of the competition which can be found online at `http://ida.first.fraunhofer.de/projects/bci/competition_ii/results/index.html` (Blankertz et al., 2004). These benchmark results indicate that classification of single-trial EEG is indeed a hard problem. Hence, our method works as a classifier producing a state-of-the art result in this hard data set. We estimated the SNR of this discriminating signal to SNR $\approx 10\log_{10}(P_{\|}/P_{\perp}) = -43$dB, where $P_{\|}$ is the power of the projected signal, and $P_{\perp}$ is the power in the orthogonal space. The achieved classification performance supports the validity of the bilinear weight space factorization in EEG.

4.2 Experiment II: Bilinear Discriminant Component Analysis of Real EEG

We applied the BDCA method in real EEG data which was recorded while the human subjects were stimulated with a sequence of images presented at a rate of ten images per second. Each subject attended to a computer monitor where most of the images where 'distractors', that is, of no particular interest. Rare but anticipated 'target' images were to be detected by the subject. This paradigm is also known as Rapid Serial Visual Presentation (RSVP) (Thorpe et al., 1996; Gerson et al., 2005). In this experiment the subject was instructed to omit any overt response to the targets but simply note its occurrence. The images used here were identical to those of Gerson et al. (2005). Sixty-four EEG channels were recorded at 2048Hz, re-referenced to average reference (one channel removed to obtain full row-rank data), filtered (for anti-aliasing) and downsampled to 128Hz sampling rate, and filtered again with a pass-band between 0.5Hz and 50Hz. All filtering was applied forwards and backwards in time to avoid introducing group delay. Trial matrices of dimension $(D,T) = (64,64)$ were extracted by epoching (500ms per epoch) the data in alignment with image stimulus. The number of recorded target/distractor trials was roughly $60/3000$ for training and $40/2000$ for testing, but varied slightly between subjects.

	Number of	Classifier performance (AUC)		
Subject	components R	Train	Cross-Valid.	Test
1	1 / 2	0.99 / 1.00	0.95 / 0.97	N.A.
2	1 / 2	0.95 / 0.97	0.92 / 0.93	0.84 / 0.87
3	1 / 2	0.91 / 0.95	0.84 / 0.83	0.85 / 0.92
4	1 / 2	0.96 / 0.99	0.91 / 0.93	0.83 / 0.85
5	1 / 2	0.92 / 0.95	0.88 / 0.87	0.92 / 0.92
6	1 / 2	0.80 / 0.90	0.65 / 0.74	0.67 / 0.85
mean	1 / 2	0.92 / 0.96	0.86 / 0.88	0.82 / 0.88

Table 2: Summary of classification performance of classifiers for all subjects. Classifiers with two components generally perform better than single-component classifiers in both cross-validation and test.

4.2.1 TRAINING AND RESULTS

We tuned the parameters using cross validation on the training set from a single subject, again using $R = 1$ and assuming standard electrode coordinates c.f. Section 4.1.1. The resulting set of parameter values are summarized in the second row of Table 1. For this subject, the cross validation AUC was 0.95, and corresponded to roughly 0.11 false positive rate and 0.89 true positive rate, indicating that the algorithm was successful in estimating a discriminating direction which was highly relevant for the experimental task. This finding underlines the usefulness of the method as a single-trial classifier for EEG.

Next, we trained classifiers for all subjects keeping the parameters fixed to the values in Table 1. The training, cross-validated, and test performances are reported in Table 2. In general the two-component classifiers performed slightly better than the single-component classifiers.

The **G** matrices (one for each subject) were estimated using the algorithm in Appendix B. Figure 1 shows a single BDCA component for each subject. Clearly, there are inter-subject variability in the spatial topographies and in the temporal profiles, however, all temporal profiles exhibit positive peaks at around 125ms and 300ms after target stimulus, and negative peak at around 200ms. The peak at 300ms in the temporal profile is in agreement with the conventional P300 which is typically observed with a rare target stimulus (Gerson et al., 2005; Parra et al., 2005). The early peak (here around 125ms) had likewise been reported previously for the RSVP paradigm (Thorpe et al., 1996). The spatial topographies shown in Figure 1 are rather complex. This may simply represent noise, but it is also possible that BDCA, using additional trials and $R > 2$, could decompose the complex rank-one patterns into more than two components with localized, that is, 'simpler', topography.

Two of the subjects (4 and 6) showed another interesting component with a broad spatial projection located slightly below the center on the scalp, see Figure 2. The component time courses were dominated by a 20Hz rhythm which seemed to modulate in amplitude around 200–300ms. This feature had not been reported before for this paradigm and underlined the usefulness of the method as a hypothesis generating tool. To validate the new hypothesis, we measured the single-component classification performances on the test set for each subject, that is, performed classification based only on the (subject specific) component shown in Figure 2. The test performances were 0.71 AUC

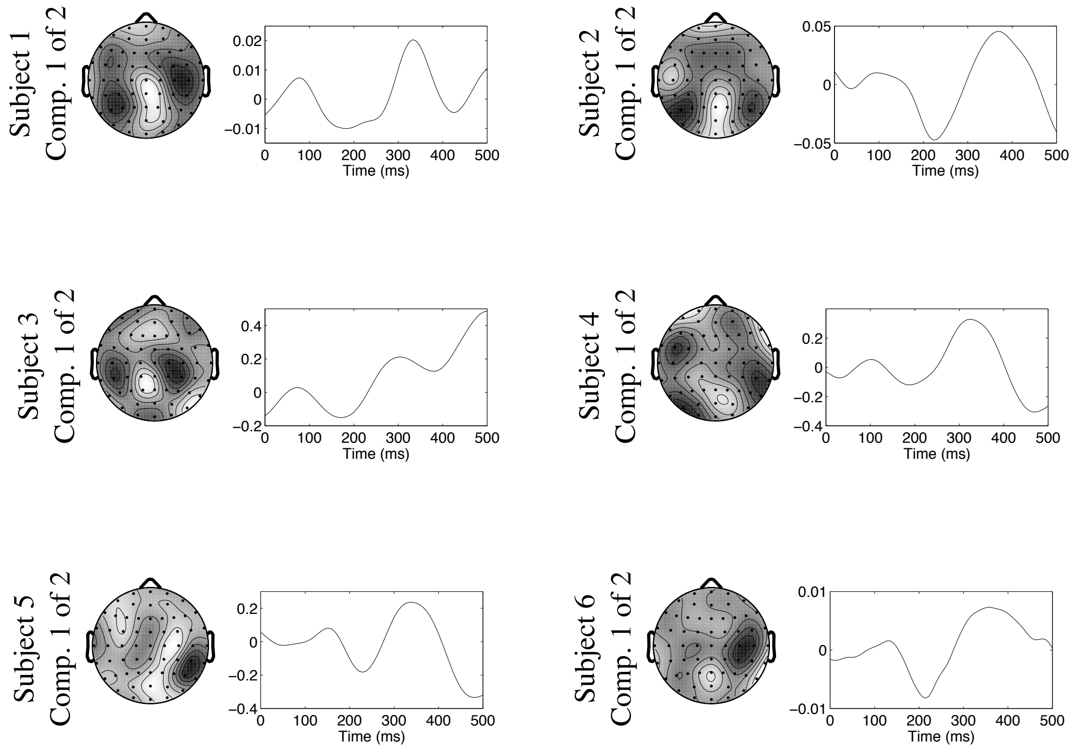


Figure 1: For each subject, one of the (spatial) $\mathbf{a}_k$ vectors is shown topographically on a cartoon head, and the corresponding (temporal) $\mathbf{b}_k$ vector is shown right next to it. The sign ambiguity between component topographies and time courses has been set so that the P300 peak has a positive projection to the center in the back of the cartoon head. All temporal profiles exhibit positive peaks at around 100ms and 350ms (P300) after target stimulus, and negative peak at around 200ms.

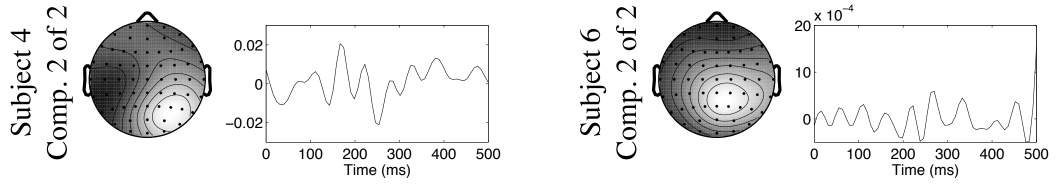


Figure 2: Two of the subjects showed a component with a broad spatial projection located slightly below center on the scalp. The component time courses shown here have that in common that they were dominated by a 20Hz rhythm which seemed to modulate in amplitude around 200–300ms.

for Subject 4, and 0.87 AUC for subject 6 which indicated that the newly identified components were indeed associated with target detection.

Finally we would like to point out that the resulting smoothness (e.g., the time courses in Figure 1) is not only affected by the choice of regularization parameters but also by the data and the number of trials. If more data is available that supports a deviation from the prior smoothness assumption the resulting time courses can and will be more punctuated in time.

5. Conclusion and Discussion

Bilinear Discriminant Analysis (BLDA) can give better classification performance in situations where a bilinear decomposition of the parameter matrix can be assumed as in (5). Such parameter matrix decompositions might prove reasonable in situations were component analysis according to model (2) is meaningful. A ‘component’ in this model is considered to be all the activity that can be associated with a common time course. One such component contributes to a rank-one subspace in data $\mathbf{X}_n$. If separate spatial distributions have separate time courses the model assumes that these contribution add up linearly. For instance, when applying this on fMRI data the implicit assumption would be that the BOLD signal originating from different neuronal populations is additive.

We presented a method for BLDA which allows smoothness regularization for better generalization performance in data sets with limited examples. This step was motivated by application to functional brain imaging were the number of examples is typically very limited compared to the data space dimensionality. The proposed BLDA method was verified in a benchmark data set, and the results were highly competitive, putting the new method on a place between the official second and third places of the competition.

We showed that BLDA can yield a data subspace factorization which makes it a useful tool for supervised extraction of components as opposed to simply a tool for classifying data matrices. We identified some essential ambiguities in such supervised subspace component decomposition and proposed to resolve them by assuming independence across the labeled mode (i.e., across trials in the EEG examples). The new method (BDCA: Bilinear Discriminant Component Analysis) thus combines BLDA with ICA and was applied to real EEG data from six human subjects. The results were in agreement with literature on the given experiment. Furthermore, a hypothesis was generated due to some components that had not been reported previously.

Model selections (i.e., finding the number of components) was based solely on classification. Then, given the optimal number of components, ICA was performed as a separate step. Future work will consider the combined likelihood to select a model that optimizes simultaneously both discrimination and independence.

Though the algorithm was motivated by functional brain imaging data (with space, time, and labeled trials as its dimensions) it should be applicable for any data set that records a matrix rather than a vector for every repetition. Examples include repetitions of spectro-temporal data such as the acoustic spectrograms of several utterances of words; multiple samples of spectro-spatial data such as multispectral images of the same areas; or spatio-temporal data such as video clips with multiple renditions of the same actions. Also, the method readily extends to the case of multiple classes or regression with continuous dependent variables. With multiple class labels one may consider multinomial logistic regression. When the dependent variables are continuous rather than discrete one can use a bilinear model with a unit link function to derive the corresponding bilinear regression. We are currently pursuing these topics in the context of EEG and fMRI.

Acknowledgments

The EEG topographies in this paper were plotted using the Matlab toolbox "EEGLAB" (Delorme and Makeig, 2004).

Appendix A. Maximum-Likelihood Estimation of U and V

Parameter estimation in a Logistic Regression model is often done though iterative maximum likelihood estimation using Newton-Raphson updates (McCulloch and Searle, 2001). In line with traditional Logistic Regression we assume the labels independent and Bernoulli distributed. The log likelihood is then given by

$$l(w_0, \{\mathbf{u}_r, \mathbf{v}_r\}) = \sum_{n=1}^{N} y_n(w_0 + \sum_{r=1}^{R} \mathbf{u}_r^{\mathrm{T}} \mathbf{X}_n \mathbf{v}_r) - \log(1 + e^{w_0 + \sum_{r=1}^{R} \mathbf{u}_r^{\mathrm{T}} \mathbf{X}_n \mathbf{v}_r}) \tag{10}$$

where y_n is the nth label, $\mathbf{u}_r$ is the rth column vector of $\mathbf{U}$, $\mathbf{v}_r$ is the rth column vector of $\mathbf{V}$, and w_0 is a scalar that enables a possible activation offset in the logistic function. Define

$$\pi(\mathbf{X}_n) \equiv \mathrm{E}[y_n] = \frac{1}{1 + e^{-(w_0 + \sum_{r=1}^{R} \mathbf{u}_r^{\mathrm{T}} \mathbf{X}_n \mathbf{v}_r)}}.$$

hen, the gradient of (10) is given by

$$\frac{\partial l}{\partial w_0} = \sum_n y_n - \pi(\mathbf{X}_n),$$

$$\frac{\partial l}{\partial \mathbf{u}_r} = \sum_n \mathbf{X}_n \mathbf{v}_r [y_n - \pi(\mathbf{X}_n)],$$

$$\frac{\partial l}{\partial \mathbf{v}_r^{\mathrm{T}}} = \sum_n \mathbf{u}_r^{\mathrm{T}} \mathbf{X}_n [y_n - \pi(\mathbf{X}_n)],$$

and the Hessian matrix entries are given by

$$\frac{\partial^2 l}{\partial w_0 \partial w_0} = -\sum_n \pi(\mathbf{X}_n)[1 - \pi(\mathbf{X}_n)],$$

$$\frac{\partial^2 l}{\partial w_0 \partial \mathbf{u}_r} = -\sum_n \mathbf{X}_n \mathbf{v}_r \pi(\mathbf{X}_n)[1 - \pi(\mathbf{X}_n)],$$

$$\frac{\partial^2 l}{\partial w_0 \partial \mathbf{v}_r^{\mathrm{T}}} = -\sum_n \mathbf{u}_r^{\mathrm{T}} \mathbf{X}_n \pi(\mathbf{X}_n)[1 - \pi(\mathbf{X}_n)],$$

$$\frac{\partial^2 l}{\partial \mathbf{u}_r \partial (\mathbf{u}_{k'})_j} = -\sum_n \mathbf{X}_n \mathbf{v}_r (\mathbf{X}_n \mathbf{v}_{k'})_j \pi(\mathbf{X}_n)[1 - \pi(\mathbf{X}_n)],$$

$$\frac{\partial^2 l}{\partial \mathbf{v}_r^{\mathrm{T}} \partial (\mathbf{v}_{k'})_j} = -\sum_n \mathbf{u}_r^{\mathrm{T}} \mathbf{X}_n (\mathbf{u}_{k'}^{\mathrm{T}} \mathbf{X}_n)_j \pi(\mathbf{X}_n)[1 - \pi(\mathbf{X}_n)],$$

$$\frac{\partial^2 l}{\partial \mathbf{u}_r \partial (\mathbf{v}_{k'}^{\mathrm{T}})_j} = \sum_n \delta_{k,k'} (\mathbf{X}_n)_{:j} [y_n - \pi(\mathbf{X}_n)] - \mathbf{X}_n \mathbf{v}_r (\mathbf{u}_{k'}^{\mathrm{T}} \mathbf{X}_n)_j \pi(\mathbf{X}_n)[1 - \pi(\mathbf{X}_n)],$$

where $\delta_{k,k'} = 1$ for $k = k'$ and zero otherwise. We provide no guarantee that the Hessian matrix will be definite, and in our experiments in this paper we obtain ML estimates using the so-called 'Damped Newton' optimization scheme which will take regularized Newton steps using adaptive regularization of the Hessian matrix (Bishop, 1996; Nielsen, 2005).

A.1 Smoothness Regularization with Gaussian Processes

The log posterior is equal to the log likelihood plus evaluation of the log prior, that is,

$$\log \mathrm{p}(w_0, \{\mathbf{u}_r, \mathbf{v}_r\} | \mathbf{X}) = l(w_0, \{\mathbf{u}_r, \mathbf{v}_r\}) + \log \mathrm{p}(w_0, \{\mathbf{u}_r, \mathbf{v}_r\}) - \log \mathrm{p}(\mathbf{X})$$

where $\mathbf{X}$ denotes data in all the trials available. Here we consider the maximum of the posterior (MAP) estimate, that is,

$$(w_0, \{\mathbf{u}_r, \mathbf{v}_r\})_{\mathrm{MAP}} = \arg \max_{w_0, \{\mathbf{u}_r, \mathbf{v}_r\}} l(w_0, \{\mathbf{u}_r, \mathbf{v}_r\}) + \log \mathrm{p}(w_0, \{\mathbf{u}_r, \mathbf{v}_r\})$$

with independent priors

$$\log \mathrm{p}(w_0, \{\mathbf{u}_r, \mathbf{v}_r\}) = \log \mathrm{p}(w_0) + \sum_r \log \mathrm{p}(\mathbf{u}_r) + \sum_r \log \mathrm{p}(\mathbf{v}_r). \tag{11}$$

For iterative MAP estimation, the terms for the Gaussian prior, to be inserted in (11), are (here shown for $\mathbf{u}_r$)

$$\log \mathrm{p}(\mathbf{u}_r) = -\frac{\dim \mathbf{u}_r}{2} \log(2\pi) - \frac{1}{2} \log(\det \mathbf{K}) - \frac{1}{2} \mathbf{u}_r^{\mathrm{T}} \mathbf{K}^{-1} \mathbf{u}_r$$

where $\dim \mathbf{u}_r = D$ (or likewise $\dim \mathbf{v}_r = T$ or $\dim w_0 = 1$) (see also Rasmussen and Williams, 2006). The extra terms, to be added to the ML terms, are; for the gradient

$$\frac{\partial \log \mathrm{p}(\mathbf{u}_r)}{\partial \mathbf{u}_r} = -\mathbf{K}^{-1} \mathbf{u}_r$$

and for the Hessian

$$\frac{\partial^2 \log \mathrm{p}(\mathbf{u}_r)}{\partial \mathbf{u}_r \partial (\mathbf{u}_r)_j} = -\mathbf{K}^{-1} \mathbf{e}_j$$

where $\mathbf{e}_j$ is the jth unit vector. These expressions (and similar for w_0 and $\mathbf{v}_r$) thus augment the terms in the maximum likelihood algorithm above.

Appendix B. Equations for Maximum-Likelihood Estimation of G

The log likelihood is given by

$$\log \mathrm{p}(\mathbf{X}_n | \hat{\mathbf{V}} \mathbf{G}^{-1}, \hat{\mathbf{U}} \mathbf{G}^{\mathrm{T}}) = -\frac{N}{2} \log \det [\mathbf{A}^{\mathrm{T}} \mathbf{A}] + \sum_n \log \mathrm{p}(\hat{\mathbf{s}}_n)$$

see also Dyrholm et al. (2006) and `http://www.imm.dtu.dk/~mad/papers/madrix.pdf`. We choose the component activation prior pdf $\mathrm{p}(\cdot) = 1/[\pi \cosh(\cdot)]$, as proposed by Bell and Sejnowski

(1995), which is appropriate for super-Gaussian independent activations (see also Lee et al., 1999). This choice however might not fit the scaling of the data very well so we parameterize the activation pdf and rewrite the likelihood

$$\log \mathrm{p}(\mathbf{X}_n|\hat{\mathbf{V}}\mathbf{G}^{-1},\hat{\mathbf{U}}\mathbf{G}^{\mathrm{T}},\alpha) = -\frac{N}{2}\log\det[\mathbf{A}^{\mathrm{T}}\mathbf{A}] + \sum_n \log \mathrm{p}(\hat{\mathbf{z}}_n/\alpha^2)$$

where $\hat{z}_k(n) = (\hat{s}_k(n) - \mathrm{E}[\hat{s}_k(n)])/\sqrt{\mathrm{var}[\hat{s}_k(n)]}$. Again, we use Damped Newton optimization which will take regularized Newton steps using adaptive regularization of the Hessian matrix (Bishop, 1996; Nielsen, 2005). We do not actually compute the Hessian but use the outer product approximation to the Hessian given by averaging gradient products across trials (see also Bishop, 1996).

The gradient of the determinant term of the log likelihood, with respect to $\mathbf{G}$, is given by

$$\frac{\partial -\frac{N}{2}\log\det[\mathbf{A}^{\mathrm{T}}\mathbf{A}]}{\partial(\mathbf{G})_{ij}} = -N\,\mathrm{Trace}\left\{(\mathbf{A}^{+})^{\mathrm{T}}\left(\frac{\partial\mathbf{A}}{\partial(\mathbf{G})_{ij}}\right)^{\mathrm{T}}\right\}$$

where

$$\frac{\partial\mathbf{A}}{\partial(\mathbf{G})_{ij}} = -\mathbf{V}(\mathbf{G}^{-1}\mathbf{e}_i\mathbf{e}_j^{\mathrm{T}}\mathbf{G}^{-1})\,|\!\otimes\!|\,\mathbf{U}\mathbf{G}^{\mathrm{T}} + \mathbf{V}\mathbf{G}^{-1}\,|\!\otimes\!|\,\mathbf{U}\mathbf{e}_j\mathbf{e}_i^{\mathrm{T}}$$

The gradient of the sum term of the log likelihood, with respect to $\mathbf{G}$, is given by

$$\frac{\partial \log \mathrm{p}(\mathbf{A}^{+}\,\mathrm{vec}(\mathbf{X}_n)/\alpha^2)}{\partial(\mathbf{G})_{ij}} = [\psi(\hat{\mathbf{s}}_n/\alpha^2)]^{\mathrm{T}}\frac{\partial\hat{\mathbf{s}}_n}{\partial(\mathbf{G})_{ij}}/\alpha^2$$

where

$$\psi(\cdot) = \frac{\mathrm{p}'(\cdot)}{\mathrm{p}(\cdot)} = -\tanh(\cdot)$$

and

$$\frac{\partial\hat{\mathbf{s}}_n}{\partial(\mathbf{G})_{ij}} = \left(\frac{\partial(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}}{\partial(\mathbf{G})_{ij}}\mathbf{A}^{\mathrm{T}} + (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\frac{\partial\mathbf{A}^{\mathrm{T}}}{\partial(\mathbf{G})_{ij}}\right)\mathrm{vec}(\mathbf{X}_n)$$

where

$$\frac{\partial(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}}{\partial(\mathbf{G})_{ij}} = -(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\,\mathrm{Trace}\left\{\frac{\partial\mathbf{A}^{\mathrm{T}}}{\partial(\mathbf{G})_{ij}}\mathbf{A} + \mathbf{A}^{\mathrm{T}}\frac{\partial\mathbf{A}}{\partial(\mathbf{G})_{ij}}\right\}(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}.$$

References

A. H. Andersen and W. S. Rayens. Structure-seeking multilinear methods for the analysis of fMRI data. *Neuroimage*, 22(2):728–39, 2004.

C.F. Beckmann and S.M. Smith. Tensorial extensions of independent component analysis for group fMRI data analysis. *NeuroImage*, 25(1):294–311, 2005.

A. J. Bell and T. J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.

C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, UK, 1996.

B. Blankertz, G. Curio, and K.-R. Müller. Classifying single trial EEG: Towards brain computer interfacing. In T. G. Diettrich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Inf. Proc. Systems 14 (NIPS 01)*, 2002.

B. Blankertz, K.-R. Müller, G. Curio, T.M. Vaughan, G. Schalk, J.R. Wolpaw, A. Schlogl, C. Neuper, G. Pfurtscheller, T. Hinterberger, M. Schroder, and N. Birbaumer. The BCI competition 2003: progress and perspectives in detection and discrimination of EEG single trials. *Biomedical Engineering, IEEE Transactions on*, 51(6):1044–1051, 2004.

R. Bro. *Multi-way Analysis in the Food Industry*. PhD thesis, Royal Veterinary and Agricultural University, Denmark, 1998.

E. T. Bullmore, S. Rabe-Hesketh, R. G. Morris, S. C. Williams, L. Gregory, J. A. Gray, and M. J. Brammer. Functional magnetic resonance image analysis of a large-scale neurocognitive network. *Neuroimage*, 4(1):16–33, 1996.

V. Calhoun, T. Adali, G. Pearlson, and J. Pekar. Spatial and temporal independent component analysis of functional MRI data containing a pair of task-related waveforms. *Hum Brain Mapp*, 13:43, 2001.

L. De Lathauwer and J. Vandewalle. Dimensionality reduction in higher-order signal processing and rank-$(R_1, R_2, ..., R_N)$ reduction in multilinear algebra. *Lin. Alg. Appl.*, 391:31–55, 2004.

A. Delorme and S. Makeig. Eeglab: an open source toolbox for analysis of single-trial eeg dynamics including independent component analysis. *J Neurosci Methods*, 134(1):9–21, 2004.

M. Dyrholm, S. Makeig, and L. K. Hansen. Model selection for convolutive ICA with an application to spatio-temporal analysis of EEG. *Neural Computation*, 2006.

M. Dyrholm and L. C. Parra. Smooth bilinear classification of EEG. In *Proceedings of the IEEE 2006 International Conference of the Engineering in Medicine and Biology Society*, 2006.

T. Fawcett. ROC graphs: Notes and practical considerations for data mining researchers. Technical report, HPL-2003-4. HP Laboratories, Palo Alto, CA, USA., 2003.

A.D. Gerson, L.C. Parra, and P. Sajda. Cortical origins of response time variability during rapid discrimination of visual objects. *NeuroImage*, 28(2):326–341, 2005.

R. A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an 'explanatory' multi-modal factor analysis. *UCLA Working Papers in Phonetics*, 16, 1970.

J. B. Kruskal. Three-way arrays: Rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and its Applications*, 18:95–138, 1977.

T.-W. Lee, M. Girolami, and T. J. Sejnowski. Independent component analysis using an extended infomax algorithm for mixed sub-gaussian and super-gaussian sources. *Neural Computation*, 11 (2):417–441, 1999.

S. Makeig, A. J. Bell, T.-P. Jung, and T. J. Sejnowski. Independent component analysis of electroencephalographic data. In M. Mozer and M. Hasselmo, editors, *Advances in Neural Information Processing Systems*, pages 145–151, 1996.

E. Martinez-Montes, P. A. Valdes-Sosa, F. Miwakeichi, R. I. Goldman, and M. S. Cohen. Concurrent EEG/fMRI analysis by multiway Partial Least Squares. *NeuroImage*, 22(3):1023–1034, 2004.

C. E. McCulloch and S. R. Searle. *Generalized, Linear, and Mixed Models*. Wiley, 2001.

F. Miwakeichi, E. Martinez-Montes, P. A. Valdes-Sosa, N. Nishiyama, H. Mizuhara, and Y. Yamaguchi. Decomposing EEG data into space-time-frequency components using parallel factor analysis. *Neuroimage*, 22(3):1035–45, 2004.

N. Mørch, L. Hansen, S. Strother, C. Svarer, D. Rottenberg, and B. Lautrup. Nonlinear vs. linear models in functional neuroimaging: Learning curves and generalization crossover. In *Proceedings of the 15th international conference on information processing in medical imaging*, volume 1230 of *Lecture Notes in Computer Science*, pages 259–270. Springer, 1997.

J. Möcks. Decomposing event-related potentials: A new topographic components model. *Biol Psychol*, 26(1-3):199–215, 1988.

M. Mørup, L. K. Hansen, C. S. Hermann, J. Parnas, and S. M. Arnfred. Parallel factor analysis as an exploratory tool for wavelet transformed event-related EEG. *NeuroImage*, 29(3):938–947, feb 2006.

H. B. Nielsen. IMMOPTIBOX. General optimization software available at `http://www.imm.dtu.dk/~hbn/immoptibox/`, 2005.

L. Parra, C. Spence, A. Gerson, and P. Sajda. Recipes for the linear analysis of EEG. *NeuroImage*, 28:326–341, 2005.

W. D. Penny, N. J. Trujillo-Barreto, and K. J. Friston. Bayesian fMRI time series analysis with spatial priors. *NeuroImage*, 24:350–362, 2005.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. 272, The MIT Press, Cambridge, Massachusetts, 2006.

Y. Sakaguchi, S. Ozawa, and M. Kotani. Feature extraction using supervised independent component analysis by maximizing class distance. In *Proc. of Int. Conf. on Neural Information Processing*, volume 5, pages 2502–2506, 2002.

K. C. Squires, E. Donchin, R. I. Herning, and G. McCarthy. On the influence of task relevance and stimulus probability on event-related-potential components. *Electroencephalogr Clin Neurophysiol*, 1977.

S. Thorpe, D. Fize, and C. Marlot. Speed of processing in the human visual system. *Nature*, 381 (6582):520–2, 1996.

M. Visani, C. Garcia, Jolion, and J.-M. Normalized radial basis function networks and bilinear discriminant analysis for face recognition. In *Proceedings. IEEE Conference on Advanced Video and Signal Based Surveillance*, pages 342–47, 2005.

J. Ye. Characterization of a family of algorithms for generalized discriminant analysis on undersampled problems. *Journal of Machine Learning Research*, 6:483–502, 2005.

Journal of Machine Learning Research 8 (2007) 1113-1143 Submitted 12/06; Published 5/07

Loop Corrections for Approximate Inference on Factor Graphs

Joris M. Mooij J.MOOIJ@SCIENCE.RU.NL
Hilbert J. Kappen B.KAPPEN@SCIENCE.RU.NL
Department of Biophysics
Radboud University Nijmegen
6525 EZ Nijmegen, The Netherlands

Editor: Michael Jordan

Abstract

We propose a method to improve approximate inference methods by correcting for the influence of loops in the graphical model. The method is a generalization and alternative implementation of a recent idea from Montanari and Rizzo (2005). It is applicable to arbitrary factor graphs, provided that the size of the Markov blankets is not too large. It consists of two steps: (i) an approximate inference method, for example, belief propagation, is used to approximate *cavity distributions* for each variable (i.e., probability distributions on the Markov blanket of a variable for a modified graphical model in which the factors involving that variable have been removed); (ii) all cavity distributions are improved by a message-passing algorithm that cancels out approximation errors by imposing certain consistency constraints. This loop correction (LC) method usually gives significantly better results than the original, uncorrected, approximate inference algorithm that is used to estimate the effect of loops. Indeed, we often observe that the loop-corrected error is approximately the square of the error of the uncorrected approximate inference method. In this article, we compare different variants of the loop correction method with other approximate inference methods on a variety of graphical models, including "real world" networks, and conclude that the LC method generally obtains the most accurate results.

Keywords: loop corrections, approximate inference, graphical models, factor graphs, belief propagation

1. Introduction

In recent years, much research has been done in the field of approximate inference on graphical models. One of the goals is to obtain accurate approximations of marginal probabilities of complex probability distributions defined over many variables, using limited computation time and memory. This research has led to a large number of approximate inference methods. Apart from sampling ("Monte Carlo") methods, there is a large number of "deterministic" approximate inference methods, such as variational methods, for example, the mean field method (Parisi, 1988), and a family of algorithms that are in some way related to the highly successful belief propagation (BP) algorithm (Pearl, 1988). BP is also known as the "sum-product algorithm" (Kschischang et al., 2001) and as "loopy belief propagation" and is directly related to the Bethe approximation (Bethe, 1935; Yedidia et al., 2005) from statistical physics. It is well-known that belief propagation yields exact results if the graphical model is a tree, or, more generally, if each connected component is a tree. If the graphical model does contain loops, BP can still yield surprisingly accurate results using little computation time. However, if the influence of loops is large, the approximate marginals calculated by BP can have large errors and the quality of the BP results may not be satisfactory.

One way to correct for the influence of short loops is to increase the cluster size of the approximation, using the cluster variation method (CVM) (Pelizzola, 2005) or other region-based approximation methods (Yedidia et al., 2005). These methods are related to the Kikuchi approximation (Kikuchi, 1951), a generalization of the Bethe approximation using larger clusters. Algorithms for calculating the CVM and related region-based approximation methods are generalized belief propagation (GBP) (Yedidia et al., 2005) and double-loop algorithms that have guaranteed convergence (Yuille, 2002; Heskes et al., 2003). By choosing the (outer) clusters such that they subsume as many loops as possible, the BP results can be improved. However, choosing a good set of outer clusters is highly nontrivial, and in general this method will only work if the clusters do not have many intersections, or in other words, if the loops do not have many intersections (see also Welling et al., 2005).

Another method that corrects for loops to a certain extent is TreeEP (Minka and Qi, 2004), a special case of expectation propagation (EP) (Minka, 2001). TreeEP does exact inference on the base tree, a subgraph of the graphical model which has no loops, and approximates the other interactions. This corrects for the loops that consist of part of the base tree and exactly one additional factor. TreeEP yields good results if the graphical model is dominated by the base tree, which is the case in very sparse models. However, loops that consist of two or more interactions that are not part of the base tree are approximated in a similar way as in BP. Hence, for denser models, the improvement of TreeEP over BP usually diminishes.

In this article we propose a method that takes into account *all* the loops in the graphical model in an approximate way and therefore obtains more accurate results in many cases. Our method is a variation on the theme introduced by Montanari and Rizzo (2005). The basic idea is to first estimate the "cavity distributions" of all variables and subsequently improve these estimates by cancelling out errors using certain consistency constraints. A *cavity distribution* of some variable is the probability distribution on its Markov blanket (all its neighboring variables) of a modified graphical model, in which all factors involving that variable have been removed. The removal of the factors breaks all the loops in which that variable takes part. This allows an approximate inference algorithm to estimate the strength of these loops in terms of effective interactions or correlations between the variables of the Markov blanket. Then, the influence of the removed factors is taken into account, which yields accurate approximations to the probability distributions of the original graphical model. Even more accuracy is obtained by imposing certain consistency relations between the cavity distributions, which results in a cancellation of errors to some extent. This error cancellation is done by a message-passing algorithm which can be interpreted as a generalization of BP in case the factor graph does not contain short loops of four nodes; indeed, assuming that the cavity distributions factorize (which they do in case there are no loops), the BP results are obtained. On the other hand, using better estimates of the effective interactions in the cavity distributions yields accurate loop-corrected results.

Although the basic idea underlying our method is very similar to that described in Montanari and Rizzo (2005), the alternative implementation that we propose here offers two advantages. Most importantly, it is directly applicable to arbitrary factor graphs, whereas the original method has only been formulated for the rather special case of graphical models with binary variables and pairwise factors, which excludes, for example, many interesting Bayesian networks. Furthermore, our implementation appears to be more robust and also gives improved results for relatively strong interactions, as will be shown numerically.

This article is organized as follows. First we explain the theory behind our proposed method and discuss the differences with the original method by Montanari and Rizzo (2005). Then we report extensive numerical experiments regarding the quality of the approximation and the computation time, where we compare with other approximate inference methods. Finally, we discuss the results and state conclusions.

2. Theory

In this work, we consider graphical models such as Markov random fields and Bayesian networks. We use the general factor graph representation since it allows for formulating approximate inference algorithms in a unified way (Kschischang et al., 2001). In the next subsection, we introduce our notation and basic definitions.

2.1 Graphical Models and Factor Graphs

Consider N discrete random variables $\{x_i\}_{i\in\mathcal{V}}$ with $\mathcal{V} := \{1,\dots,N\}$. Each variable x_i takes values in a discrete domain $\mathcal{X}_i$. We will use the following multi-index notation: for any subset $I \subseteq \mathcal{V}$, we write $x_I := (x_{i_1}, x_{i_2}, \dots, x_{i_m})$ if $I = \{i_1, i_2, \dots, i_m\}$ and $i_1 < i_2 < \dots i_m$. We consider a probability distribution over $x = (x_1,\dots,x_N)$ that can be written as a product of factors ψ_I:

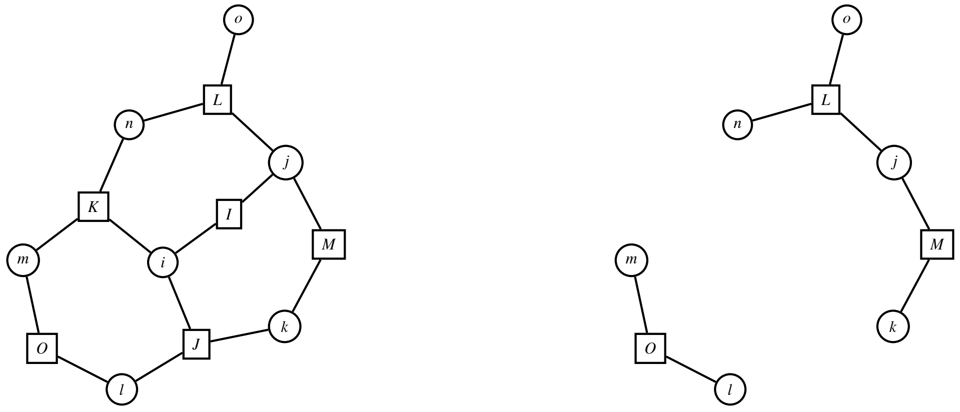
$$P(x) = \frac{1}{Z}\prod_{I\in\mathcal{F}} \psi_I(x_I), \qquad Z = \sum_x \prod_{I\in\mathcal{F}} \psi_I(x_I). \tag{1}$$

The factors (which we will also call "interactions") are indexed by (small) subsets of $\mathcal{V}$, that is, $\mathcal{F} \subseteq \mathcal{P}(\mathcal{V}) := \{I : I \subseteq \mathcal{V}\}$. Each factor is a nonnegative function $\psi_I : \prod_{i\in I}\mathcal{X}_i \to [0,\infty)$. For a Bayesian network, the factors are conditional probability tables. In case of Markov random fields, the factors are often called potentials (not to be confused with statistical physics terminology, where "potential" refers to minus the logarithm of the factor instead). Henceforth, we will refer to a triple $(\mathcal{V}, \mathcal{F}, \{\psi_I\}_{I\in\mathcal{F}})$ that satisfies the description above as a discrete *graphical model* (or *network*).

In general, the normalizing constant Z is not known and exact computation of Z is infeasible, due to the fact that the number of terms to be summed is exponential in N. Similarly, computing marginal distributions $P(x_J)$ of P for subsets of variables $J \subseteq \mathcal{V}$ is intractable in general. In this article, we focus on the task of accurately approximating single-variable marginals $P(x_i) = \sum_{x_{\mathcal{V}\setminus\{i\}}} P(x)$.

We can represent the structure of the probability distribution (1) using a *factor graph*. This is a bipartite graph, consisting of *variable nodes* $i \in \mathcal{V}$ and *factor nodes* $I \in \mathcal{F}$, with an edge between i and I if and only if $i \in I$, that is, if x_i participates in the factor ψ_I. We will represent factor nodes visually as rectangles and variable nodes as circles. See Figure 1(a) for an example of a factor graph. We denote the neighboring nodes of a variable node i by $N_i := \{I \in \mathcal{F} : i \in I\}$ and the neighboring nodes of a factor node I simply by $I = \{i \in \mathcal{V} : i \in I\}$. Further, we define for each variable $i \in \mathcal{V}$ the set $\Delta i := \bigcup N_i$ consisting of all variables that appear in some factor in which variable i participates, and the set $\partial i := \Delta i \setminus \{i\}$, the *Markov blanket* of i.

In the following, we will often abbreviate the set theoretical notation $X \setminus Y$ (i.e., all elements in X that are not in Y) by $\setminus Y$ if it is obvious from the context what the set X is. Also, we will write $X \setminus y$ instead of $X \setminus \{y\}$. Further, we will use lowercase for variable indices and uppercase for factor



(a) Original factor graph (b) Cavity graph of i

Figure 1: (a) Original factor graph, corresponding to the probability distribution $P(x) = \frac{1}{Z}\psi_L(x_j,x_n,x_o)\psi_I(x_i,x_j)\psi_M(x_j,x_k)\psi_K(x_i,x_m,x_n)\psi_J(x_i,x_k,x_l)\psi_O(x_l,x_m)$; (b) Factor graph corresponding to the cavity network of variable i, obtained by removing variable i and the factor nodes that contain i (i.e., I, J and K). The Markov blanket of i is $\partial i = \{j,k,l,m,n\}$. The cavity distribution $Z^{\setminus i}(x_{\partial i})$ is the (unnormalized) marginal on $x_{\partial i}$ of the probability distribution corresponding to the cavity graph (b).

indices. For convenience, we will define for any subset $\mathcal{A} \subset \mathcal{F}$ the product of the corresponding factors:

$$\Psi_{\mathcal{A}}(x_{\bigcup \mathcal{A}}) := \prod_{I \in \mathcal{A}} \psi_I(x_I).$$

2.2 Cavity Networks and Loop Corrections

The notion of a *cavity* stems from statistical physics, where it was used originally to calculate properties of random ensembles of certain graphical models (Mézard et al., 1987). A cavity is obtained by removing one variable from the graphical model, together with all the factors in which that variable participates.

In our context, we define cavity networks as follows (see also Figure 1):

Definition 2.1 *Given a graphical model* $(\mathcal{V}, \mathcal{F}, \{\psi_I\}_{I\in\mathcal{F}})$ *and a variable* $i \in \mathcal{V}$, *the* cavity network *of variable i is the graphical model* $(\mathcal{V} \setminus i, \mathcal{F} \setminus N_i, \{\psi_I\}_{I\in\mathcal{F}\setminus N_i})$.

The probability distribution corresponding to the cavity network of variable i is thus proportional to:

$$\Psi_{\setminus N_i}(x_{\setminus i}) = \prod_{\substack{I\in\mathcal{F} \\ i \notin I}} \psi_I(x_I).$$

Summing out all the variables, except for the neighbors ∂i of i, gives what we will call the *cavity distribution*:

Definition 2.2 *Given a graphical model* $(\mathcal{V}, \mathcal{F}, \{\psi_I\}_{I\in\mathcal{F}})$ *and a variable* $i \in \mathcal{V}$*, the* cavity distribution *of* i *is*

$$Z^{\setminus i}(x_{\partial i}) := \sum_{x_{\setminus \Delta i}} \Psi_{\setminus N_i}(x_{\setminus i}). \tag{2}$$

Thus the cavity distribution of i is proportional to the marginal of the cavity network of i on the Markov blanket ∂i. The cavity distribution describes the *effective* interactions (or correlations) induced by the cavity network on the neighbors ∂i of variable i. Indeed, from Equations (1) and (2) and the trivial observation that $\Psi_{\mathcal{F}} = \Psi_{N_i}\Psi_{\setminus N_i}$ we conclude:

$$P(x_{\Delta i}) \propto Z^{\setminus i}(x_{\partial i})\Psi_{N_i}(x_{\Delta i}). \tag{3}$$

Thus, given the cavity distribution $Z^{\setminus i}(x_{\partial i})$, one can calculate the marginal distribution of the original graphical model P on $x_{\Delta i}$, provided that the cardinality of $\mathcal{X}_{\Delta i}$ is not too large.

In practice, exact cavity distributions are not known, and the only way to proceed is to use approximate cavity distributions. Given some approximate inference method (e.g., BP), there are two ways to calculate $P(x_{\Delta i})$: either use the method to approximate $P(x_{\Delta i})$ directly, or use the method to approximate $Z^{\setminus i}(x_{\partial i})$ and use Equation (3) to obtain an approximation to $P(x_{\Delta i})$. The latter approach generally gives more accurate results, since the complexity of the cavity network is less than that of the original network. In particular, the cavity network of variable i contains no loops involving that variable, since all factors in which i participates have been removed (e.g., the loop $i - J - l - O - m - K - i$ in the original network, Figure 1(a), is not present in the cavity network, Figure 1(b)). Thus the latter approach to calculating $P(x_{\Delta i})$ takes into account loops involving variable i, although in an approximate way. It does not, however, take into account the other loops in the original graphical model. The basic idea of the loop correction approach of Montanari and Rizzo (2005) is to use the latter approach for all variables in the network, but to adjust the approximate cavity distributions in order to cancel out approximation errors before (3) is used to obtain the final approximate marginals. This approach takes into account *all* the loops in the original network, in an approximate way.

This basic idea can be implemented in several ways. Here we propose an implementation which we will show to have certain advantages over the original implementation proposed in Montanari and Rizzo (2005). In particular, it is directly applicable to arbitrary factor graphs with variables taking an arbitrary (discrete) number of values and factors that may contain zeroes and consist of an arbitrary number of variables. In the remaining subsections, we will first discuss our proposed implementation in detail. In Section 2.6 we will discuss differences with the original approach.

2.3 Combining Approximate Cavity Distributions to Cancel Out Errors

Suppose that we have obtained an initial approximation $\zeta_0^{\setminus i}(x_{\partial i})$ of the (exact) cavity distribution $Z^{\setminus i}(x_{\partial i})$, for each $i \in \mathcal{V}$. Let $i \in \mathcal{V}$ and consider the approximation error of the cavity distribution of i, that is, the exact cavity distribution of i divided by its approximation:

$$\frac{Z^{\setminus i}(x_{\partial i})}{\zeta_0^{\setminus i}(x_{\partial i})}.$$

In general, this is an arbitrary function of the variables $x_{\partial i}$. However, for our purposes, we *approximate* the error as a product of factors defined on small subsets of ∂i in the following way:

$$\frac{Z^{\setminus i}(x_{\partial i})}{\zeta_0^{\setminus i}(x_{\partial i})} \approx \prod_{I \in N_i} \phi_I^{\setminus i}(x_{I \setminus i}).$$

Thus we assume that the approximation error lies near a submanifold parameterized by the error factors $\{\phi_I^{\setminus i}(x_{I\setminus i})\}_{I \in N_i}$. If we were able to calculate these error factors, we could improve our initial approximation $\zeta_0^{\setminus i}(x_{\partial i})$ by replacing it with the product

$$\zeta^{\setminus i}(x_{\partial i}) := \zeta_0^{\setminus i}(x_{\partial i}) \prod_{I \in N_i} \phi_I^{\setminus i}(x_{I \setminus i}) \approx Z^{\setminus i}(x_{\partial i}). \tag{4}$$

Using (3), this would then yield an improved approximation of $P(x_{\Delta i})$.

It turns out that the error factors can indeed be calculated by exploiting the redundancy of the information in the initial cavity approximations $\{\zeta_0^{\setminus i}\}_{i \in \mathcal{V}}$. The fact that all $\zeta^{\setminus i}$ provide approximations to marginals of the *same* probability distribution $P(x)$ via (3) can be used to obtain consistency constraints. The number of constraints obtained in this way is usually enough to solve for the unknown error factors $\{\phi_I^{\setminus i}(x_{I\setminus i})\}_{i \in \mathcal{V}, I \in N_i}$.

Here we propose the following consistency constraints. Let $Y \in \mathcal{F}$, $i \in Y$ and $j \in Y$ with $i \neq j$ (see also Figure 2). Consider the graphical model $(\mathcal{V}, \mathcal{F} \setminus Y, \{\psi_I\}_{I \in \mathcal{F} \setminus Y})$ that is obtained from the original graphical model by removing factor ψ_Y. The product of all factors (except ψ_Y) obviously satisfies:

$$\Psi_{\setminus Y} = \Psi_{N_i \setminus Y} \Psi_{\setminus N_i} = \Psi_{N_j \setminus Y} \Psi_{\setminus N_j}.$$

Using (2) and summing over all x_k for $k \notin Y \setminus i$, we obtain the following equation, which holds for the exact cavity distributions $Z^{\setminus i}$ and $Z^{\setminus j}$:

$$\sum_{x_i} \sum_{x_{\Delta i \setminus Y}} \Psi_{N_i \setminus Y} Z^{\setminus i} = \sum_{x_i} \sum_{x_{\Delta j \setminus Y}} \Psi_{N_j \setminus Y} Z^{\setminus j}.$$

Substituting our basic assumption (4) on both sides and pulling the factor $\phi_Y^{\setminus i}(x_{Y \setminus i})$ in the l.h.s. through the summation, we obtain:

$$\phi_Y^{\setminus i} \sum_{x_i} \sum_{x_{\Delta i \setminus Y}} \Psi_{N_i \setminus Y} \zeta_0^{\setminus i} \prod_{I \in N_i \setminus Y} \phi_I^{\setminus i} = \sum_{x_i} \sum_{x_{\Delta j \setminus Y}} \Psi_{N_j \setminus Y} \zeta_0^{\setminus j} \prod_{J \in N_j} \phi_J^{\setminus j}.$$

Since this should hold for each $j \in Y \setminus i$, we can take the geometric mean of the r.h.s. over all $j \in Y \setminus i$. After rearranging, this yields:

$$\phi_Y^{\setminus i} = \frac{\left(\prod_{j \in Y \setminus i} \sum_{x_i} \sum_{x_{\Delta j \setminus Y}} \Psi_{N_j \setminus Y} \zeta_0^{\setminus j} \prod_{J \in N_j} \phi_J^{\setminus j} \right)^{1/|Y \setminus i|}}{\sum_{x_i} \sum_{x_{\Delta i \setminus Y}} \Psi_{N_i \setminus Y} \zeta_0^{\setminus i} \prod_{I \in N_i \setminus Y} \phi_I^{\setminus i}} \qquad \text{for all } i \in \mathcal{V}, Y \in N_i. \tag{5}$$

Note that the numerator is an approximation of the joint marginal $P^{\setminus Y}(x_{Y \setminus i})$ of the modified graphical model $(\mathcal{V}, \mathcal{F} \setminus Y, \{\psi_I\}_{I \in \mathcal{F} \setminus Y})$ on the variables $Y \setminus i$.

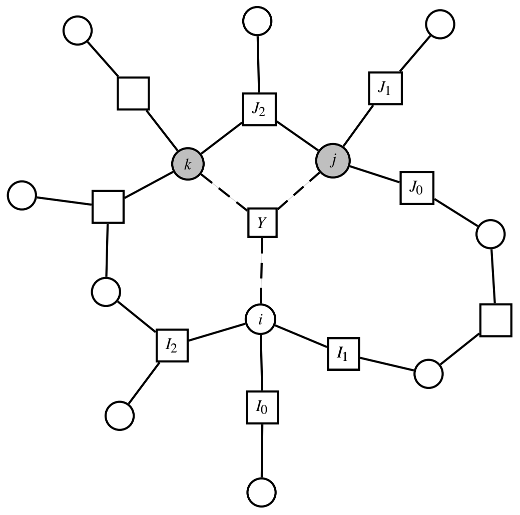


Figure 2: Part of the factor graph, illustrating the derivation of (5). The two gray variable nodes correspond to $Y \setminus i = \{j,k\}$.

Solving the consistency Equations (5) simultaneously for the error factors $\{\phi_I^{\setminus i}\}_{i\in\mathcal{V},I\in N_i}$ can be done using a simple fixed point iteration algorithm, for example, Algorithm 1. The input consists of the initial approximations $\{\zeta_0^{\setminus i}\}_{i\in\mathcal{V}}$ to the cavity distributions. It calculates the error factors that satisfy (5) by fixed point iteration and from the fixed point, it calculates improved approximations of the cavity distributions $\{\zeta^{\setminus i}\}_{i\in\mathcal{V}}$ using Equation (4).[1] From the improved cavity distributions, the loop-corrected approximations to the single-variable marginals of the original probability distribution (1) can be calculated as follows:

$$P_i(x_i) \approx b_i(x_i) \propto \sum_{x_{\partial i}} \Psi_{N_i}(x_{\Delta i})\zeta^{\setminus i}(x_{\partial i}), \tag{6}$$

where the factor ψ_Y is now included. Algorithm 1 uses a sequential update scheme, but other update schemes are possible (e.g., random sequential or parallel). In practice, the fixed sequential update scheme often converges without the need for damping.

Alternatively, one can formulate Algorithm 1 in terms of the "beliefs"

$$Q_i(x_{\Delta i}) \propto \Psi_{N_i}(x_{\Delta i})\zeta_0^{\setminus i}(x_{\partial i}) \prod_{I\in N_i} \phi_I^{\setminus i}(x_{I\setminus i}) = \Psi_{N_i}(x_{\Delta i})\zeta^{\setminus i}(x_{\partial i}). \tag{7}$$

As one easily verifies, the update equation

$$Q_i \leftarrow Q_i \frac{\prod_{j\in Y\setminus i} \left(\sum_{x_{\Delta j\setminus(Y\setminus i)}} Q_j \psi_Y^{-1} \right)^{1/|Y\setminus i|}}{\sum_{x_{\Delta i\setminus(Y\setminus i)}} Q_i \psi_Y^{-1}}$$

1. Alternatively, one could formulate the updates directly in terms of the cavity distributions $\{\zeta^{\setminus i}\}$.

Algorithm 1 Loop Correction Algorithm

```
Input:   initial approximate cavity distributions $\{\zeta_0^{\setminus i}\}_{i\in\mathcal{V}}$
Output:  improved approximate cavity distributions $\{\zeta^{\setminus i}\}_{i\in\mathcal{V}}$

 1: repeat
 2:   for all $i \in \mathcal{V}$ do
 3:     for all $Y \in N_i$ do
 4:       $\phi_Y^{\setminus i}(x_{Y\setminus i}) \leftarrow \frac{\left(\prod_{j\in Y\setminus i}\sum_{x_i}\sum_{x_{\Delta j\setminus Y}}\Psi_{N_j\setminus Y}\zeta_0^{\setminus j}\prod_{J\in N_j}\phi_J^{\setminus j}\right)^{1/|Y\setminus i|}}{\sum_{x_i}\sum_{x_{\Delta i\setminus Y}}\Psi_{N_i\setminus Y}\zeta_0^{\setminus i}\prod_{I\in N_i\setminus Y}\phi_I^{\setminus i}}$
 5:     end for
 6:   end for
 7: until convergence
 8: for all $i \in \mathcal{V}$ do
 9:   $\zeta^{\setminus i}(x_{\partial i}) \leftarrow \zeta_0^{\setminus i}(x_{\partial i})\prod_{I\in N_i}\phi_I^{\setminus i}(x_{I\setminus i})$
10: end for
```

is equivalent to line 1 of Algorithm 1. Intuitively, the update improves the approximate distribution Q_i on Δi by replacing its marginal on $Y \setminus i$ (in the absence of Y) by a more accurate approximation of this marginal, namely the numerator. Written in this form, the algorithm is reminiscent of iterative proportional fitting (IPF). However, contrary to IPF, the desired marginals are also updated at each iteration. Note that after convergence, the large beliefs $Q_i(x_{\Delta i})$ need not be consistent, that is, in general $\sum_{x_{\Delta i\setminus J}} Q_i \neq \sum_{x_{\Delta j\setminus J}} Q_j$ for $i, j \in \mathcal{V}, J \subseteq \Delta i \cap \Delta j$.

2.4 A Special Case: Factorized Cavity Distributions

In the previous subsection we have discussed how to improve approximations of cavity distributions. We now discuss what happens when we use the simplest possible initial approximations $\{\zeta_0^{\setminus i}\}_{i\in\mathcal{V}}$, namely constant functions, in Algorithm 1. This amounts to the assumption that no loops are present. We will show that if the factor graph does not contain short loops consisting of four nodes, fixed points of the standard BP algorithm are also fixed points of Algorithm 1. In this sense, Algorithm 1 can be considered to be a generalization of the BP algorithm. In fact, this holds even if the initial approximations factorize in a certain way, as will be shown below.

If all factors involve at most two variables, one can easily arrange for the factor graph to have no loops of four nodes. See Figure 1(a) for an example of a factor graph which has no loops of four nodes. The factor graph depicted in Figure 2 does have a loop of four nodes: $k - Y - j - J_2 - k$.

Theorem 2.1 *If the factor graph corresponding to (1) has no loops of exactly four nodes, and all initial approximate cavity distributions factorize in the following way:*

$$\zeta_0^{\setminus i}(x_{\partial i}) = \prod_{I\in N_i}\xi_I^{\setminus i}(x_{I\setminus i}) \qquad \forall i \in \mathcal{V}, \tag{8}$$

then fixed points of the BP algorithm can be mapped to fixed points of Algorithm 1. Furthermore, the corresponding variable and factor marginals obtained from (7) are identical to the BP beliefs.

Proof Note that replacing the initial cavity approximations by

$$\zeta_0^{\setminus i}(x_{\partial i}) \mapsto \zeta_0^{\setminus i}(x_{\partial i}) \prod_{I \in N_i} \varepsilon_I^{\setminus i}(x_{I \setminus i})$$

for arbitrary positive functions $\varepsilon_I^{\setminus i}(x_{I\setminus i})$ does not change the beliefs (7) corresponding to the fixed points of (5). Thus, without loss of generality, we can assume $\zeta_0^{\setminus i}(x_{\partial i}) = 1$ for all $i \in \mathcal{V}$. The BP update equations are (Kschischang et al., 2001):

$$\begin{aligned} \mu_{j\to I}(x_j) &\propto \prod_{J \in N_j \setminus I} \mu_{J \to j}(x_j) && j \in \mathcal{V}, I \in N_j, \\ \mu_{I \to i}(x_i) &\propto \sum_{x_{I\setminus i}} \psi_I(x_I) \prod_{j \in I \setminus i} \mu_{j \to I}(x_j) && I \in \mathcal{F}, i \in I \end{aligned} \tag{9}$$

in terms of messages $\{\mu_{J\to j}(x_j)\}_{j\in\mathcal{V}, J \in N_j}$ and $\{\mu_{j\to J}(x_j)\}_{j\in\mathcal{V}, J \in N_j}$. Assume that the messages μ are a fixed point of (9) and take the *Ansatz*

$$\phi_I^{\setminus i}(x_{I\setminus i}) = \prod_{k \in I \setminus i} \mu_{k \to I}(x_k) \qquad \text{for } i \in \mathcal{V}, I \in N_i.$$

Then, for $i \in \mathcal{V}$, $Y \in N_i$, $j \in Y \setminus i$, we can write out part of the numerator of (5) as follows:

$$\begin{aligned} \sum_{x_i} \sum_{x_{\Delta j \setminus Y}} \Psi_{N_j \setminus Y} \zeta_0^{\setminus j} \prod_{J \in N_j} \phi_J^{\setminus j} &= \sum_{x_i} \sum_{x_{\Delta j \setminus Y}} \phi_Y^{\setminus j} \prod_{J \in N_j \setminus Y} \psi_J \phi_J^{\setminus j} \\ &= \sum_{x_i} \left(\prod_{k \in Y \setminus j} \mu_{k \to Y} \right) \prod_{J \in N_j \setminus Y} \sum_{x_{J \setminus j}} \psi_J \prod_{k \in J \setminus j} \mu_{k \to J} \\ &= \sum_{x_i} \left(\prod_{k \in Y \setminus j} \mu_{k \to Y} \right) \mu_{j \to Y} = \sum_{x_i} \prod_{k \in Y} \mu_{k \to Y} \propto \prod_{k \in Y \setminus i} \mu_{k \to Y} \\ &= \phi_Y^{\setminus i}, \end{aligned}$$

where we used the BP update Equations (9) and rearranged the summations and products using the assumption that the factor graph has no loops of four nodes. Thus, the numerator of the r.h.s. of (5) is simply $\phi_Y^{\setminus i}$. Using a similar calculation, one can derive that the denominator of the r.h.s. of (5) is constant, and hence (5) is valid (up to an irrelevant constant).

For $Y \in \mathcal{F}$, $i \in Y$, the marginal on x_Y of the belief (7) can be written in a similar way:

$$\begin{aligned} \sum_{x_{\Delta i \setminus Y}} Q_i \propto \sum_{x_{\Delta i \setminus Y}} \Psi_{N_i} \prod_{I \in N_i} \phi_I^{\setminus i} &= \sum_{x_{\Delta i \setminus Y}} \prod_{I \in N_i} \psi_I \prod_{k \in I \setminus i} \mu_{k \to I} \\ &= \psi_Y \left(\prod_{k \in Y \setminus i} \mu_{k \to Y} \right) \prod_{I \in N_i \setminus Y} \sum_{x_{I \setminus i}} \psi_I \prod_{k \in I \setminus i} \mu_{k \to I} \\ &= \psi_Y \left(\prod_{k \in Y \setminus i} \mu_{k \to Y} \right) \prod_{I \in N_i \setminus Y} \mu_{I \to i} = \psi_Y \left(\prod_{k \in Y \setminus i} \mu_{k \to Y} \right) \mu_{i \to Y} \\ &= \psi_Y \prod_{k \in Y} \mu_{k \to Y}, \end{aligned}$$

which is proportional to the BP belief $b_Y(x_Y)$ on x_Y. Hence, also the single-variable marginal b_i defined in (6) corresponds to the BP single-variable belief, since both are marginals of b_Y for $Y \in N_i$. ■

If the factor graph does contain loops of four nodes, we usually observe that the fixed point of Algorithm 1 coincides with the solution of the "minimal" CVM approximation when using factorized initial cavity approximations as in (8). The minimal CVM approximation uses all maximal factors as outer clusters (a *maximal* factor is a factor defined on a domain which is not a strict subset of the domain of another factor). In that case, the factor beliefs found by Algorithm 1 are consistent, that is, $\sum_{x_{\Delta i \setminus Y}} Q_i = \sum_{x_{\Delta j \setminus Y}} Q_j$ for $i, j \in Y$, and are identical to the minimal CVM factor beliefs. In particular, this holds for all the graphical models used in Section 3.[2]

2.5 Obtaining Initial Approximate Cavity Distributions

There is no principled way to obtain the initial cavity approximations $\zeta_0^{\setminus i}(x_{\partial i})$. In the previous subsection, we investigated the results of applying the LC algorithm on factorizing initial cavity approximations. More sophisticated approximations that do take into account the effect of loops can significantly enhance the accuracy of the final result. Here, we will describe one method, which uses BP on clamped cavity networks. This method captures all interactions in the cavity distribution of i in an approximate way and can lead to very accurate results. Instead of BP, any other approximate inference method that gives an approximation of the normalizing constant Z in (1) can be used, such as mean field, TreeEP (Minka and Qi, 2004), a double-loop version of BP (Heskes et al., 2003) which has guaranteed convergence towards a minimum of the Bethe free energy, or some variant of GBP (Yedidia et al., 2005). One could also choose the method for each cavity separately, trading accuracy versus computation time. We focus on BP because it is a very fast and often relatively accurate algorithm.

Let $i \in \mathcal{V}$ and consider the cavity network of i. For each possible state of $x_{\partial i}$, run BP on the cavity network clamped to that state $x_{\partial i}$ and calculate the corresponding Bethe free energy $F_{Bethe}^{\setminus i}(x_{\partial i})$ (Yedidia et al., 2005). Then, take the following initial approximate cavity distribution:

$$\zeta_0^{\setminus i}(x_{\partial i}) \propto e^{-F_{Bethe}^{\setminus i}(x_{\partial i})}.$$

This procedure is exponential in the size of ∂i: it uses $\prod_{j \in \partial i} |\mathcal{X}_j|$ BP runs. However, many networks encountered in applications are relatively sparse and have limited cavity size and the computational cost may be acceptable.

This particular way of obtaining initial cavity distributions has the following interesting property: in case the factor graph contains only a single loop and assuming that the fixed point is unique, the final beliefs (7) resulting from Algorithm 1 are exact. This can be shown using an argument similar to that given in Montanari and Rizzo (2005). Suppose that the graphical model contains exactly one loop and let $i \in \mathcal{V}$. First, consider the case that i is part of the loop; removing i will break the loop and the remaining cavity network will be singly connected. The cavity distribution approximated by BP will thus be exact. Now if i is not part of the loop, removing i will divide the

2. In a draft version of this work (Mooij and Kappen, 2006), we conjectured that the result of Algorithm 1, when initialized with factorizing initial cavity approximations, would *always* coincide with the minimal CVM approximation. This conjecture no longer stands because we have found a counter example.

network into several connected components, one for each neighbor of i. This implies that the cavity distribution calculated by BP contains no higher-order interactions, that is, $\zeta_0^{\setminus i}$ is exact modulo single-variable interactions. Because the final beliefs (7) are invariant under perturbation of the $\zeta_0^{\setminus i}$ by single-variable interactions, the final beliefs calculated by Algorithm 1 are exact if the fixed point is unique.

If all interactions are pairwise and each variable is binary and has exactly $|\partial i| = d$ neighbors, the time complexity of the resulting "loop-corrected BP" (LCBP) algorithm is given by $O(N2^d EI_{BP} + Nd2^{d+1}I_{LC})$, where E is the number of edges in the factor graph, I_{BP} is the average number of iterations of BP on a clamped cavity network and I_{LC} is the number of iterations needed to obtain convergence in Algorithm 1.

2.6 Differences with the Original Implementation

As mentioned before, the idea of estimating the cavity distributions and imposing certain consistency relations amongst them has been first presented in Montanari and Rizzo (2005). In its simplest form (i.e., the so-called first-order correction), the implementation of that basic idea as proposed by Montanari and Rizzo (2005) differs from our proposed implementation in the following aspects.

First, the original method described by Montanari and Rizzo (2005) is only formulated for the rather special case of binary variables and pairwise interactions. In contrast, our method is formulated in a general way that makes it applicable to factor graphs with variables having more than two possible values and factors consisting of more than two variables. Also, factors may contain zeroes. The generality that our implementation offers is important for many practical applications. In the rest of this section, we will assume that the graphical model (1) belongs to the special class of models with binary variables with pairwise interactions, allowing further comparison of both implementations.

An important difference is that Montanari and Rizzo (2005) suggest to deform the initial approximate cavity distributions by altering certain *cumulants* (also called "connected correlations"), instead of altering certain interactions. In general, for a set $\mathcal{A}$ of ± 1-valued random variables $\{x_i\}_{i\in\mathcal{A}}$, one can define for any subset $\mathcal{B} \subseteq \mathcal{A}$ the *moment*

$$M_{\mathcal{B}} := \sum_{x_{\mathcal{A}}} P(x_{\mathcal{A}}) \prod_{j\in\mathcal{B}} x_j.$$

The moments $\{M_{\mathcal{B}}\}_{\mathcal{B}\subseteq\mathcal{A}}$ are a parameterization of the probability distribution $P(x_{\mathcal{A}})$. An alternative parameterization is given in terms of the cumulants. The *(joint) cumulants* $\{C_{\mathcal{E}}\}_{\mathcal{E}\subseteq\mathcal{A}}$ are certain polynomials of the moments, defined implicitly by the following equations:

$$M_{\mathcal{B}} = \sum_{C\in\mathrm{Part}(\mathcal{B})} \prod_{\mathcal{E}\in C} C_{\mathcal{E}}$$

where Part($\mathcal{B}$) is the set of partitions of $\mathcal{B}$.[3] In particular, $C_i = M_i$ and $C_{ij} = M_{ij} - M_i M_j$ for all $i, j \in \mathcal{A}$ with $i \neq j$. Montanari and Rizzo (2005) propose to approximate the cavity distributions by estimating the pair cumulants and assuming higher-order cumulants to be zero. Then, the singleton cumulants (i.e., the single-variable marginals) are altered, keeping higher-order cumulants fixed, in

3. For a set X, a *partition* of X is a nonempty set Y such that each $Z \in Y$ is a nonempty subset of X and $\bigcup Y = X$.

such a way as to impose consistency of the single-variable marginals, in the absence of interactions shared by two neighboring cavities. We refer the reader to Appendix A for a more detailed description of the implementation in terms of cumulants suggested by Montanari and Rizzo (2005).

The assumption suggested in Montanari and Rizzo (2005) that higher-order cumulants are zero is the most important difference with our method, which instead takes into account effective interactions in the cavity distribution of *all* orders. In principle, the cumulant parameterization also allows for taking into account higher-order cumulants, but this would not be very efficient due to the combinatorics needed for handling the partitions.

A minor difference lies in the method to obtain initial approximations to the cavity distributions. Montanari and Rizzo (2005) propose to use BP in combination with linear response theory to obtain the initial pairwise cumulants. This difference is not very important, since one could also use BP on clamped cavity networks instead, which turns out to give almost identical results.

As we will show in Section 3, our method of altering interactions appears to be more robust and still works in regimes with strong interactions, whereas the cumulant implementation suffers from convergence problems for strong interactions.

Montanari and Rizzo (2005) also derive a linearized version of their cumulant-based scheme (by expanding up to first order in terms of the pairwise cumulants, see Appendix A) which is quadratic in the size of the cavity. This linearized, cumulant-based version is currently the only one that can be applied to networks with large Markov blankets (cavities), that is, where the maximum number of states $\max_{i\in\mathcal{V}} |\mathcal{X}_{\Delta i}|$ is large, provided that all variables are binary and interactions are pairwise.

3. Numerical Experiments

We have performed various numerical experiments to compare the quality of the results and the computation time of the following approximate inference methods:

MF Mean field, with a random sequential update scheme and no damping.

BP Belief propagation. We have used the recently proposed update scheme (Elidan et al., 2006), which converges also for difficult problems without the need for damping.

TreeEP TreeEP (Minka and Qi, 2004), without damping. We generalized the method of choosing the base tree described in Minka and Qi (2004) to multiple variable factors as follows: when estimating the mutual information between x_i and x_j, we take the product of the marginals on $\{i, j\}$ of all the factors that involve x_i and/or x_j. Other generalizations of TreeEP to higher-order factors are possible (e.g., by clustering variables), but it is not clear how to do this in general in an optimal way.

LCBP ("Loop-corrected belief propagation") Algorithm 1, where the approximate cavities are initialized according to the description in Section 2.5.

LCBP-Cum The original cumulant-based loop correction scheme by Montanari and Rizzo (2005), using response propagation (also known as linear response) to approximate the initial pairwise cavity cumulants. The full update Equations (14) are used and higher-order cumulants are assumed to vanish. For strong interactions, the update Equations (14) often yield values for the $\mathcal{M}_j^{\setminus i}$ outside of the valid interval $[-1, 1]$. In this case, we project these values back into the valid interval in the hope that the method will converge to a valid result, which it sometimes does.

LCBP-Cum-Lin Similar to LCBP-Cum, but instead of the full update Equations (14), the linearized update Equations (15) are used.

CVM-Min A double-loop implementation (Heskes et al., 2003) of the minimal CVM approximation, which uses (maximal) factors as outer clusters.

CVM-Δ A double-loop implementation of CVM using the sets $\{\Delta i\}_{i\in\mathcal{V}}$ as outer clusters. These are the same sets of variables as used by LCBP (c.f. (7)) and therefore it is interesting to compare both algorithms.

CVM-Loop*k* A double-loop implementation of CVM, using as outer clusters all (maximal) factors together with all loops in the factor graph that consist of up to k different variables (for $k = 3,4,5,6,8$).

We have used a double-loop implementation of CVM instead of GBP because the former is guaranteed to converge to a local minimum of the Kikuchi free energy (Heskes et al., 2003) without damping, whereas the latter often only converges with strong damping. The difficulty with damping is that the optimal damping constant is not known *a priori*, which necessitates multiple trial runs with different damping constants, until a suitable one is found. Using too much damping slows down convergence, whereas a certain amount of damping is required to obtain convergence in the first place. Therefore, in general we expect that (damped) GBP is not much faster than a double-loop implementation because of the computational cost of finding the optimal damping constant.

To be able to assess the errors of the various approximate methods, we have only considered problems for which exact inference (using a standard JunctionTree method) was still feasible.

For each approximate inference method, we report the maximum ℓ_∞ error of the approximate single-variable marginals b_i, calculated as follows:

$$\text{Error} := \max_{i\in\mathcal{V}} \max_{x_i\in\mathcal{X}_i} |b_i(x_i) - P(x_i)|$$

where $P(x_i)$ is the exact marginal calculated using the JunctionTree method.

The computation time was measured as CPU time in seconds on a 2.4 GHz AMD Opteron 64bits processor with 4 GB memory. The timings should be seen as indicative because we have not spent equal amounts of effort optimizing each method.[4]

We consider an iterative method to be "converged" after T time steps if for each variable $i \in \mathcal{V}$, the ℓ_∞ distance between the approximate probability distributions of that variable at time step T and $T+1$ is less than $\varepsilon = 10^{-9}$.

We have studied four different model classes: (i) random graphs of uniform degree with pairwise interactions and binary variables; (ii) random factor graphs with binary variables and factor nodes of uniform degree $k = 3$; (iii) the ALARM network, which has variables taking on more than two possible values and factors consisting of more than two variables; (iv) PROMEDAS networks, which have binary variables but factors consisting of more than two variables. For more extensive experiments, see Mooij and Kappen (2006).

4. Our C++ implementation of various approximate inference algorithms is free/open source software and can be downloaded from `http://www.mbfys.ru.nl/~jorism/libDAI`.

3.1 Random Regular Graphs with Binary Variables

We have compared various approximate inference methods on random graphs, consisting of N binary (± 1-valued) variables, having only pairwise interactions, where each variable has the same degree $|\partial i| = d$. In this case, the probability distribution (1) can be written in the following way:

$$P(x) = \frac{1}{Z} \exp\left(\sum_{i \in \mathcal{V}} \theta_i x_i + \frac{1}{2} \sum_{i \in \mathcal{V}} \sum_{j \in \partial i} J_{ij} x_i x_j \right).$$

The parameters $\{\theta_i\}_{i \in \mathcal{V}}$ are called the *local fields* and the parameters $\{J_{ij} = J_{ji}\}_{i \in \mathcal{V}, j \in \partial i}$ are called the *couplings*. The graph structure and the parameters θ and J were drawn randomly for each instance. The local fields $\{\theta_i\}$ were drawn independently from a $\mathcal{N}(0, \beta\Theta)$ distribution (i.e., a normal distribution with mean 0 and standard deviation $\beta\Theta$). For the couplings $\{J_{ij}\}$, we took mixed ("spin-glass") couplings, drawn independently from a normal distribution $J_{ij} \sim \mathcal{N}\left(0, \beta \tanh^{-1} \frac{1}{\sqrt{d-1}}\right)$. The constant β (called "inverse temperature" in statistical physics) controls the overall interaction strength and thereby the difficulty of the inference problem, larger β corresponding usually to more difficult problems. The constant Θ controls the relative strength of the local fields, where larger Θ result in easier inference problems. The particular d-dependent scaling of the couplings is used in order to obtain roughly d-independent behavior. For $\Theta = 0$ and for $\beta \approx 1$, a phase transition occurs in the limit of $N \to \infty$, going from an easy "paramagnetic" phase for $\beta < 1$ to a complicated "spin-glass" phase for $\beta > 1$.[5]

We have also done experiments with positive ("attractive" or "ferromagnetic") couplings, but the conclusions from these experiments did not differ significantly from those using mixed couplings (Mooij and Kappen, 2006). Therefore we do not report those experiments here.

3.1.1 $N = 100, d = 3$, STRONG LOCAL FIELDS ($\Theta = 2$)

We have studied various approximate inference methods on regular random graphs of low degree $d = 3$, consisting of $N = 100$ variables, with relatively strong local fields of strength $\Theta = 2$. We have considered various overall interaction strengths β between 0.01 and 10. For each value of β, we have used 16 random instances. On each instance, we have run various approximate inference algorithms.

Figure 3 shows results for MF, BP and TreeEP, and their loop-corrected versions, LCMF, LCBP and LCTreeEP. The loop-corrected versions are the result of Algorithm 1, initialized with approximate cavity distributions obtained by the procedure described in Section 2.5 (using MF, BP, and TreeEP in the role of BP). Note that the loop correction method significantly reduces the error in each case. In fact, on average the loop-corrected error is approximately given by the square of the uncorrected error, as is apparent from the scatter plots in Figure 4. BP is the fastest of the uncorrected methods and TreeEP is the most accurate but also the slowest uncorrected method. MF is both slower and less accurate than BP. Unsurprisingly, the loop-corrected methods show similar relative performance behaviors. Because BP is very fast and relatively accurate, we focus on LCBP in the rest of this article. Note further that although the graph is rather sparse, the improvement of LCBP over BP is significantly more than the improvement of TreeEP over BP.

5. More precisely, the PA-SG phase transition occurs at $\Theta = 0$ and $(d-1) = \langle \tanh^2(\beta J_{ij}) \rangle$, where $\langle \cdot \rangle$ is the average over all J_{ij} (Mooij and Kappen, 2005). What happens for $\Theta > 0$ is not known, to the best of our knowledge.

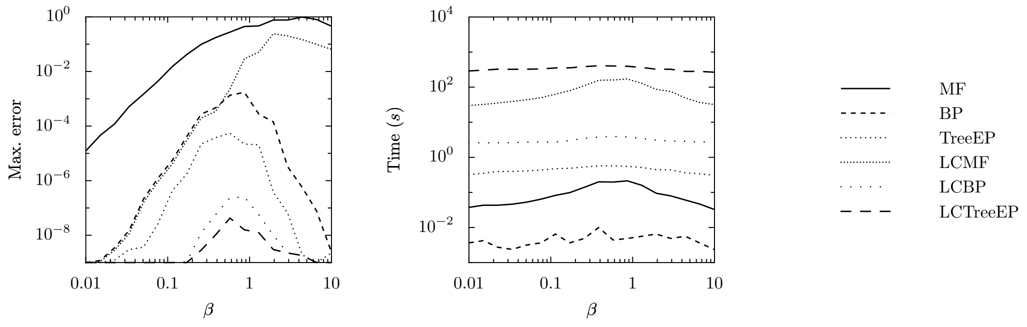


Figure 3: Error (left) and computation time (right) as a function of interaction strength for various approximate inference methods (MF, BP, TreeEP) and their loop-corrected versions (LCMF, LCBP, LCTreeEP). The averages (calculated in the logarithmic domain) were computed from the results for 16 randomly generated instances of $(N = 100, d = 3)$ regular random graphs with strong local fields $\Theta = 2$.

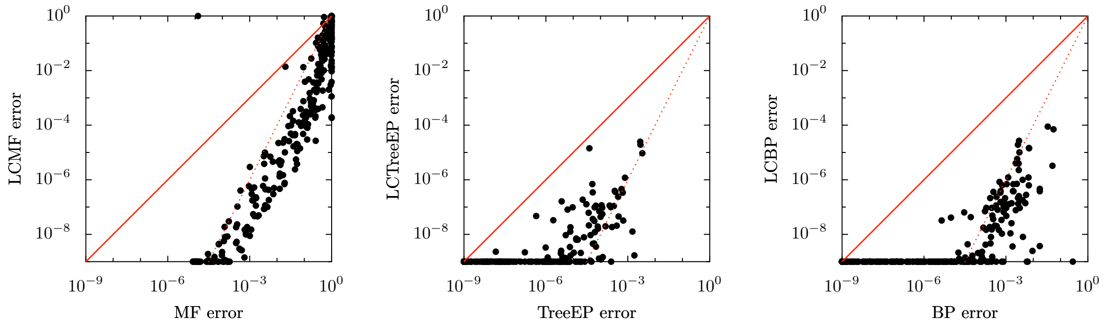


Figure 4: Pairwise comparisons of errors of uncorrected and loop-corrected methods, for the same instances as in Figure 3. The solid red lines correspond with $y = x$, the dotted red lines with $y = x^2$. Only the cases have been plotted for which both approximate inference methods have converged. Saturation of errors around 10^{-9} is an artifact due to the convergence criterion.

In Figures 5 and 6 we compare the different implementations of the loop correction method on the same instances as used before. For small values of β, LCBP-Cum and LCBP-Cum-Lin both converge and yield high quality results, and the error introduced by the linearization is relatively small. However, for larger values of β, both methods get more and more convergence problems, although for the few cases where they do converge, they still yield accurate results. At $\beta \approx 10$, both methods have completely stopped converging. The error introduced by the linearization increases for larger values of β. The computation times of LCBP-Cum, LCBP-Cum-Lin and LCBP do not differ substantially in the regime where all methods converge. However, the quality of the LCBP results is higher than that of the cumulant-based methods. This is mainly due to the fact that LCBP also takes into account effective triple interactions in the initial estimates of the approximate cavity distributions.

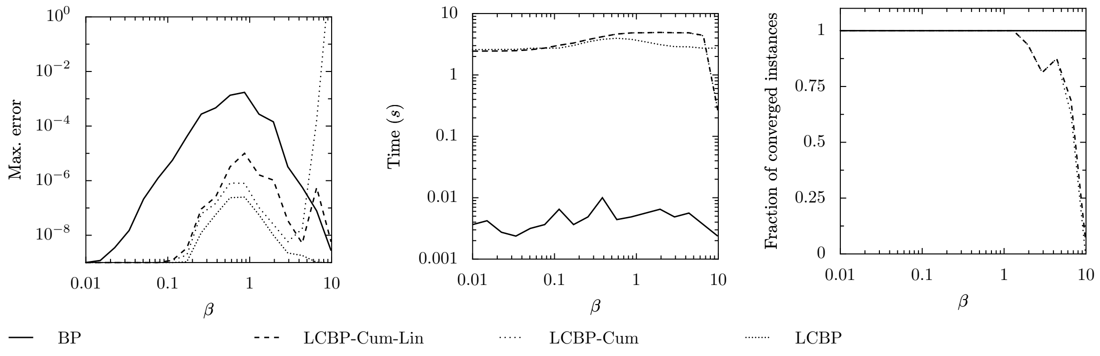


Figure 5: For the same instances as in Figure 3: average error (left), average computation time (center) and fraction of converged instances (right) as a function of interaction strength β for various variants of the LC method. The averages of errors and computation time were calculated from the converged instances only. The average computation time and fraction of converged instances for LCBP-Cum and LCBP-Cum-Lin are difficult to distinguish, because they are (almost) identical.

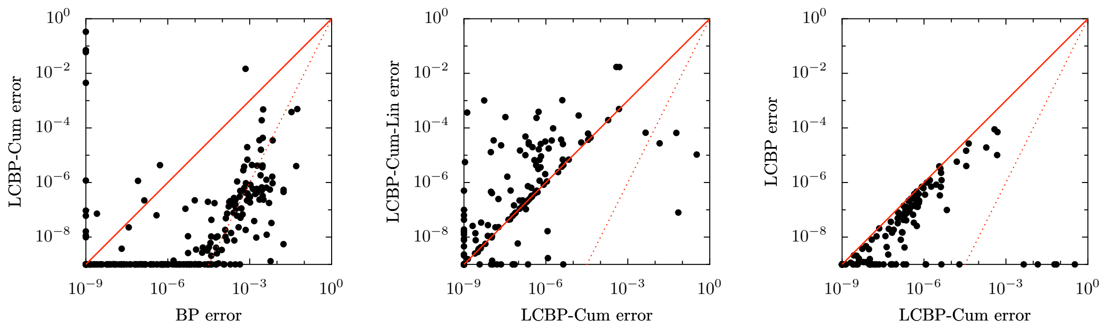


Figure 6: Pairwise comparisons of errors of various methods for the same instances as in Figure 3. Only the cases have been plotted for which both approximate inference methods converged.

We speculate that the reason for the break-down of LCBP-Cum and LCBP-Cum-Lin for strong interactions is due to the choice of cumulants instead of interactions. Indeed, consider two random variables x_1 and x_2 with fixed pair interaction $\exp(Jx_1x_2)$. By altering the singleton interactions $\exp(\theta_1 x_1)$ and $\exp(\theta_2 x_2)$, one can obtain any desired marginals of x_1 and x_2. However, a fixed pair cumulant $C_{12} = \langle x_1x_2\rangle - \langle x_1\rangle\langle x_2\rangle$ imposes a constraint on the range of possible expectation values $\langle x_1\rangle$ and $\langle x_2\rangle$ (hence on the single-variable marginals of x_1 and x_2); the freedom of choice in these marginals becomes less as the pair cumulant becomes stronger. We believe that something similar happens for LCBP-Cum (and LCBP-Cum-Lin): for strong interactions, the approximate pair cumulants in the cavity are strong, and even tiny errors can lead to inconsistencies which prevent convergence.

The results of the CVM approach to loop correction are shown in Figures 7 and 8. The CVM-Loop methods, with clusters reflecting the short loops present in the factor graph, do indeed improve

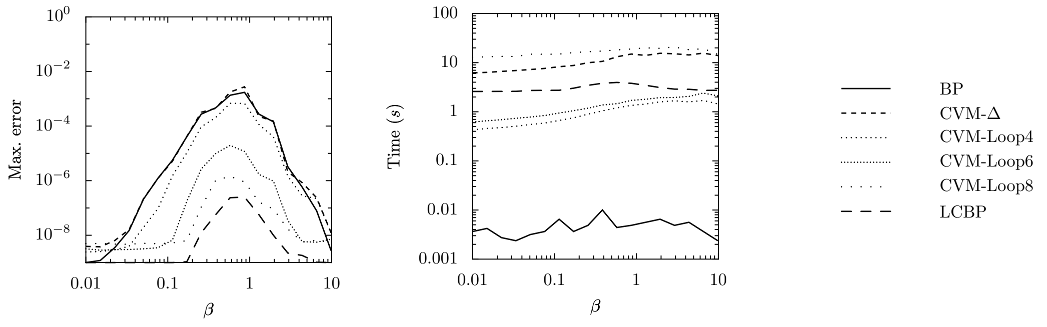


Figure 7: Average errors (left) and computation times (right) for various CVM methods (and LCBP, for reference) on the same instances as in Figure 3. All methods converged on all instances.

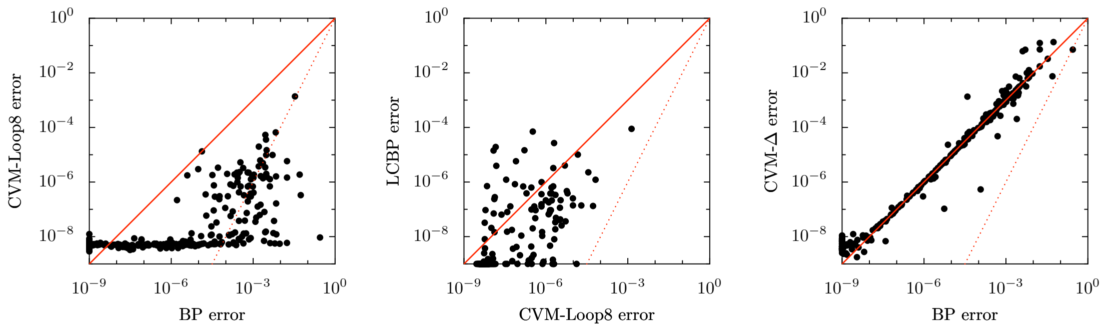


Figure 8: Pairwise comparisons of errors for various methods for the same instances as in Figure 3.

on BP. Furthermore, as expected, the use of larger clusters (that subsume longer loops) improves the results, although computation time quickly increases. CVM-Loop3 (not plotted) turned out not to give any improvement over BP, simply because there were (almost) no loops of 3 variables present. The most accurate CVM method, CVM-Loop8, needs more computation time than LCBP, whereas it yields inferior results.[6]

In addition to the CVM-Loop methods, we compared with the CVM-Δ method, which uses $\{\Delta i\}_{i\in\mathcal{V}}$ as outer clusters. These clusters subsume the clusters used implicitly by BP (which are simply the pairwise factors) and therefore one would naively expect that the CVM-Δ approximation yields better results. Surprisingly however, the quality of CVM-Δ is *similar* to that of BP, although its computation time is enormous. This illustrates that simply using larger clusters for CVM does not always lead to better results. Furthermore, we conclude that although LCBP and CVM-Δ use identical clusters to approximate the target probability distribution, the nature of both approximations is very different.

6. The CVM errors are often seen to saturate around 10^{-8}, which indicates that the slow convergence of the CVM double-loop algorithm in these cases requires a stricter convergence criterion.

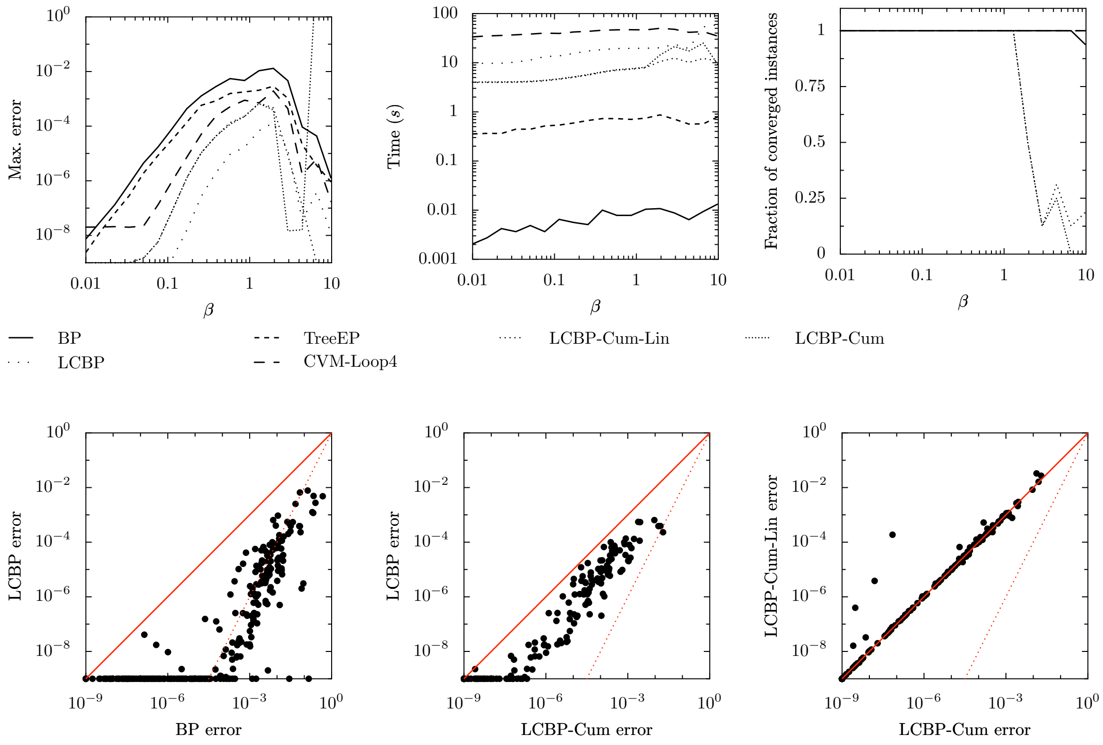


Figure 9: Selected results for $(N = 50, d = 6)$ regular random graphs with strong local fields $\Theta = 2$. The averaged results for LCBP-Cum and LCBP-Cum-Lin nearly coincide for $\beta \lesssim 1$.

3.1.2 WEAK LOCAL FIELDS ($\Theta = 0.2$)

We have done the same experiments also for weak local fields ($\Theta = 0.2$), with the other parameters unaltered (i.e., $N = 100$, $d = 3$). The picture roughly remains the same, apart from the following differences. First, the influence of the phase transition is more pronounced; many methods have severe convergence problems around $\beta = 1$. Second, the negative effect of linearization on the error (LCBP-Cum-Lin compared to LCBP-Cum) is smaller.

3.1.3 LARGER DEGREE ($d = 6$)

To study the influence of the degree $d = |\partial i|$, we have done additional experiments for $d = 6$. We had to reduce the number of variables to $N = 50$, because exact inference was infeasible for larger values of N due to quickly increasing treewidth. The results are shown in Figure 9. As in the previous experiments, BP is the fastest and least accurate method, whereas LCBP yields the most accurate results, even for high β. Again we see that the LCBP error is approximately the square of the BP error and that LCBP gives better results than LCBP-Cum, but needs more computation time.

However, we also note the following differences with the case of low degree ($d = 3$). The relative improvement of TreeEP over BP has decreased. This could have been expected, because in denser networks, the effect of taking out a tree becomes less.

Further, the relative improvement of CVM-Loop4 over BP has increased, probably because there are more short loops present. On the other hand, computation time of CVM-Loop4 has also

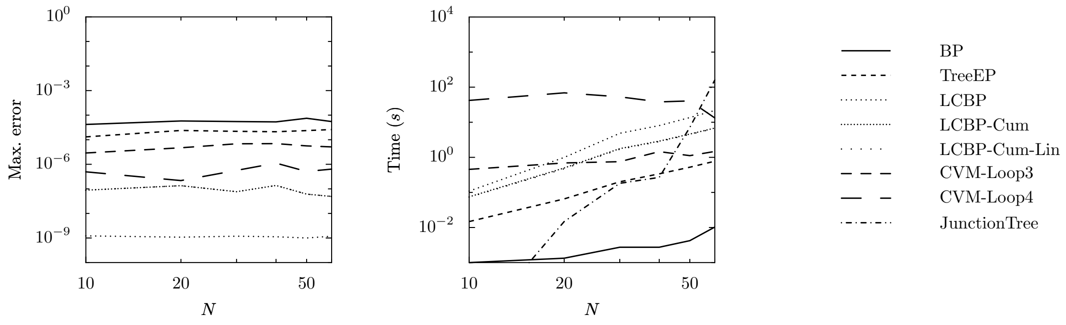


Figure 10: Error (left) and computation time (right) as a function of N (the number of variables), for random graphs with uniform degree $d = 6$, $\beta = 0.1$ and $\Theta = 2$. Points are averages over 16 randomly generated instances. Each method converged on all instances. The results for LCBP-Cum and LCBP-Cum-Lin coincide.

increased and it is the slowest of all methods. We decided to abort the calculations for CVM-Loop6 and CVM-Loop8, because computation time was prohibitive due to the enormous amount of short loops present. We conclude that the CVM-Loop approach to loop correction is not very efficient if there are many loops present.

Surprisingly, the results of LCBP-Cum-Lin are now very similar in quality to the results of LCBP-Cum, except for a few isolated cases (presumably on the edge of the convergence region).

3.1.4 SCALING WITH N

We have investigated how computation time and error scale with the number of variables N, for fixed $\beta = 0.1$, $\Theta = 2$ and $d = 6$. We used a machine with more memory (16 GB) to be able to do exact inference without swapping also for $N = 60$. The results are shown in Figure 10. The error of each method is approximately constant.

BP computation time should scale approximately linearly in N, which is difficult to see in this plot. LCBP variants are expected to scale quadratic in N (since d is fixed) which we have verified by checking the slopes of corresponding lines in the plot for large values of N. The computation time of CVM-Loop3 and CVM-Loop4 seems to be approximately constant, probably because the large number of overlaps of short loops for small values of N causes difficulties. The computation time of the exact JunctionTree method quickly increases due to increasing treewidth; for $N = 60$ it is already ten times larger than the computation time of the slowest approximate inference method.

We conclude that for large N, exact inference is infeasible, whereas LCBP still yields very accurate results using moderate computation time.

3.1.5 SCALING WITH d

It is also interesting to see how various methods scale with d, the variable degree, which is directly related to the cavity size. We have done experiments for random graphs of size $N = 24$ with fixed $\beta = 0.1$ and $\Theta = 2$ for different values of d between 3 and 23. The results can be found in Figure 11. We aborted the calculations of the slower methods (LCBP, LCBP-Cum, CVM-Loop3) at $d = 15$.

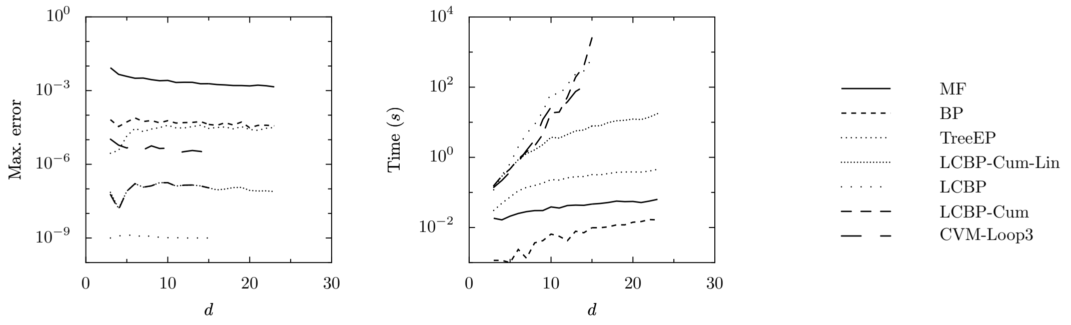


Figure 11: Error (left) and computation time (right) as a function of variable degree d for regular random graphs of $N = 24$ variables for $\beta = 0.1$ and $\Theta = 2$. Points are averages over 16 randomly generated instances. Each method converged on all instances. Errors of LCBP-Cum and LCBP-Cum-Lin coincide for $d \leq 15$; for $d > 15$, LCBP-Cum became too slow.

Due to the particular dependence of the interaction strength on d, the errors of most methods depend only slightly on d. TreeEP is an exception: for larger d, the relative improvement of TreeEP over BP diminishes, and the TreeEP error approaches the BP error. CVM-Loop3 gives better quality, but needs relatively much computation time and becomes very slow for large d due to the large increase in the number of loops of 3 variables. LCBP is the most accurate method, but becomes very slow for large d. LCBP-Cum is less accurate and becomes slower than LCBP for large d, because of the additional overhead of the combinatorics needed to perform the update equations. The accuracy of LCBP-Cum-Lin is indistinguishable from that of LCBP-Cum, although it needs significantly less computation time.

Overall, we conclude from Section 3.1 that for these binary, pairwise graphical models, LCBP is the best method for obtaining high accuracy marginals if the graphs are sparse, LCBP-Cum-Lin is the best method if the graphs are dense and LCBP-Cum shows no clear advantages over either method.

3.2 Multi-variable Factors

We now go beyond pairwise interactions and study a class of random factor graphs with binary variables and uniform factor degree $|I| = k$ (for all $I \in \mathcal{F}$) with $k > 2$. The number of variables is N and the number of factors is M. The factor graphs are constructed by starting from an empty graphical model $(\mathcal{V}, \emptyset, \emptyset)$ and adding M random factors, where each factor is obtained in the following way: a subset $I = \{I_1, \dots, I_k\} \subseteq \mathcal{V}$ of k different variables is drawn; a vector of 2^k independent random numbers $\{J_I(x_I)\}_{x_I \in \mathcal{X}_I}$ is drawn from a $\mathcal{N}(0, \beta)$ distribution; the factor $\psi_I(x_I) := \exp J_I(x_i)$ is added to the graphical model. We only use those constructed factor graphs that are connected.[7] The parameter β again controls the interaction strength.

We have done experiments for $(N = 50, M = 50, k = 3)$ for various values of β between 0.01 and 2. For each value of β, we have used 16 random instances. For higher values of β, computation

7. The reason that we require the factor graph to be connected is that not all our approximate inference method implementations currently support connected factor graphs that consist of more than one connected component.

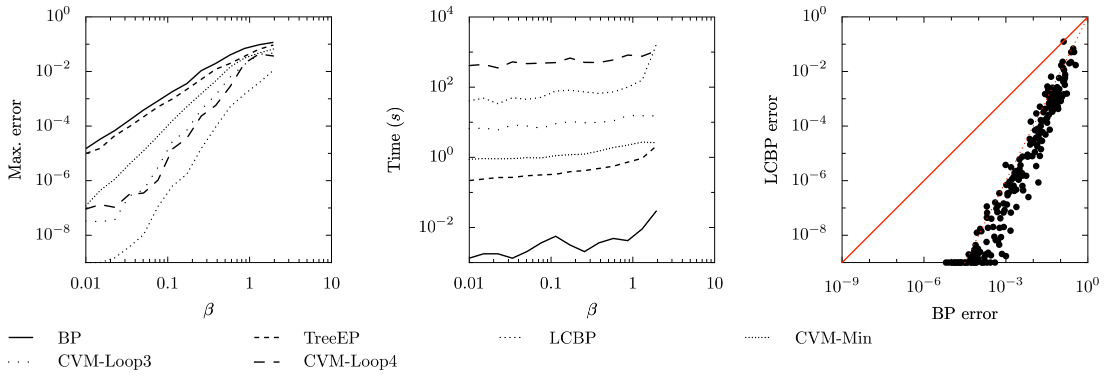


Figure 12: Results for $(N = 50, M = 50, k = 3)$ random factor graphs.

times increased quickly and convergence became problematic for BP, TreeEP and LCBP. This is probably related to the effects of a phase transition. The results are shown in Figure 12.

Looking at the error and the computation time in Figure 12, the following ranking can be made, where accuracy and computation time both increase: BP, TreeEP, CVM-Min, CVM-Loop3, LCBP. CVM-Loop4 uses more computation time than LCBP but gives worse results. LCBP-Cum and LCBP-Cum-Lin are not available due to the fact that the factors involve more than two variables. Note that the improvement of TreeEP over BP is rather small. Further, note that the LCBP error is again approximately given by the square of the BP error.

3.3 ALARM Network

The ALARM network[8] is a well-known Bayesian network consisting of 37 variables (some of which can take on more than two possible values) and 37 factors (many of which involve more than two variables). In addition to the usual approximate inference methods, we have compared with GBP-Min, a GBP implementation of the minimal CVM approximation that uses maximal factors as outer clusters. The results are reported in Table 1.[9]

The accuracy of GBP-Min (and CVM-Min) is almost identical to that of BP for this graphical model; GBP-Min converges without damping and is faster than CVM-Min. On the other hand, TreeEP significantly improves the BP result in roughly the same time as GBP-Min needs. Simply enlarging the cluster size (CVM-Δ) slightly deteriorates the quality of the results and also causes an enormous increase of computation time. The quality of the CVM-Loop results is roughly comparable to that of TreeEP. Surprisingly, increasing the loop depth beyond 4 deteriorates the quality of the results and results in an explosion of computation time. We conclude that the CVM-Loop method is not a very good approach to correcting loops in this case. LCBP uses considerable computation time, but yields errors that are approximately 10^4 times smaller than BP errors. The cumulant-

8. The ALARM network can be downloaded from `http://compbio.cs.huji.ac.il/Repository/Datasets/alarm/alarm.dsc`.

9. In Mooij et al. (2007), we also report experimental results for the ALARM network. In that work, we used another update rule for LCBP, which explains the different error obtained there ($5.4 \cdot 10^{-04}$). The update rule (5) used in the present work generally yields better results for higher-order interactions, whereas for pairwise interactions, both update rules are equivalent.

Method	Time (s)	Error
BP	0.00	$2.026 \cdot 10^{-01}$
TreeEP	0.21	$3.931 \cdot 10^{-02}$
GBP-Min	0.18	$2.031 \cdot 10^{-01}$
CVM-Min	1.13	$2.031 \cdot 10^{-01}$
CVM-Δ	280.67	$2.233 \cdot 10^{-01}$
CVM-Loop3	1.19	$4.547 \cdot 10^{-02}$
CVM-Loop4	154.97	$3.515 \cdot 10^{-02}$
CVM-Loop5	1802.83	$5.316 \cdot 10^{-02}$
CVM-Loop6	84912.70	$5.752 \cdot 10^{-02}$
LCBP	23.67	$3.412 \cdot 10^{-05}$

Table 1: Results for the ALARM network

based loop LCBP methods are not available, due to the presence of factors involving more than two variables and variables that can take more than two values.

3.4 PROMEDAS Networks

In this subsection, we study the performance of LCBP on another "real world" example, the PROMEDAS medical diagnostic network (Wiegerinck et al., 1999). The diagnostic model in PROMEDAS is based on a Bayesian network. The global architecture of this network is similar to QMR-DT (Shwe et al., 1991). It consists of a diagnosis layer that is connected to a layer with findings.[10] Diagnoses (diseases) are modeled as *a priori* independent binary variables causing a set of symptoms (findings), which constitute the bottom layer. The PROMEDAS network currently consists of approximately 2000 diagnoses and 1000 findings.

The interaction between diagnoses and findings is modeled with a noisy-OR structure. The conditional probability of the finding given the parents is modeled by $m+1$ numbers, m of which represent the probabilities that the finding is caused by one of the diseases and one that the finding is not caused by any of the parents.

The noisy-OR conditional probability tables with m parents can be naively stored in a table of size 2^m. This is problematic for the PROMEDAS networks since findings that are affected by more than 30 diseases are not uncommon in the PROMEDAS network. We use an efficient implementation of noisy-OR relations as proposed by Takikawa and D'Ambrosio (1999) to reduce the size of these tables. The trick is to introduce dummy variables s and to make use of the property

$$\mathrm{OR}(x|y_1,y_2,y_3) = \sum_s \mathrm{OR}(x|y_1,s)\mathrm{OR}(s|y_2,y_3).$$

The factors on the right hand side involve at most 3 variables instead of the initial 4 (left). Repeated application of this formula reduces all factors to triple interactions or smaller.

When a patient case is presented to PROMEDAS, a subset of the findings will be clamped and the rest will be unclamped. If our goal is to compute the marginal probabilities of the diagnostic

10. In addition, there is a layer of variables, such as age and gender, that may affect the prior probabilities of the diagnoses. Since these variables are always clamped for each patient case, they merely change the prior disease probabilities and are irrelevant for our current considerations.

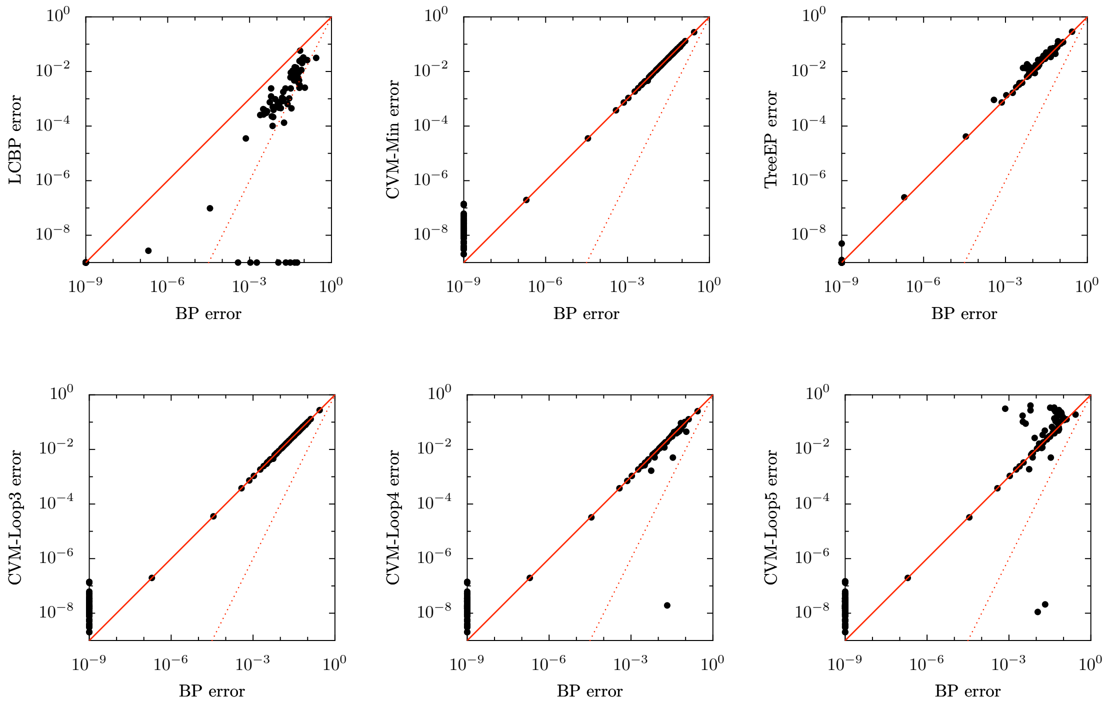


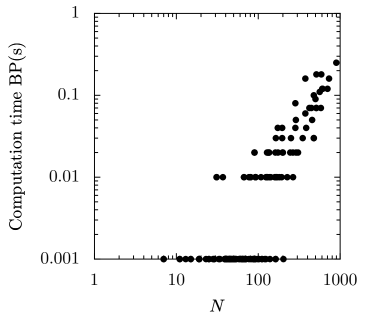
Figure 13: Scatter plots of errors for PROMEDAS instances.

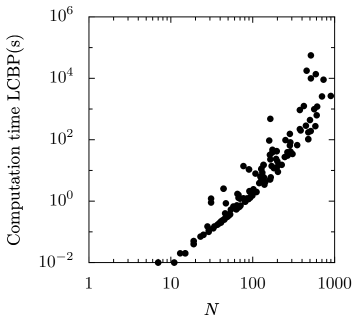
variables only, the unclamped findings and the diagnoses that are not related to any of the clamped findings can be summed out of the network as a preprocessing step. The clamped findings cause an effective interaction between their parents. However, the noisy-OR structure is such that when the finding is clamped to a negative value, the effective interaction factorizes over its parents. Thus, findings can be clamped to negative values without additional computation cost (Jaakkola and Jordan, 1999).

The complexity of the problem now depends on the set of findings that is given as input. The more findings are clamped to a positive value, the larger the remaining network of disease variables and the more complex the inference task. Especially in cases where findings share more than one common possible diagnosis, and consequently loops occur, the model can become complex.

We use the PROMEDAS model to generate virtual patient data by first clamping one of the disease variables to be positive and then clamping each finding to its positive value with probability equal to the conditional distribution of the finding, given the positive disease. The union of all positive findings thus obtained constitute one patient case. For each patient case, the corresponding truncated graphical model is generated. The number of disease nodes in this truncated graph is typically quite large.

The results can be found in Figures 13 and 14. Surprisingly, neither TreeEP nor any of the CVM methods gives substantial improvements over BP. TreeEP even gives worse results compared to BP. The CVM-Min and CVM-Loop3 results appear to be almost identical to the BP results. CVM-Loop4 manages to improve over BP in a few cases. Increased loop depth ($k = 5, 6$) results in worse quality in many cases and also in an enormous increase in computation time.





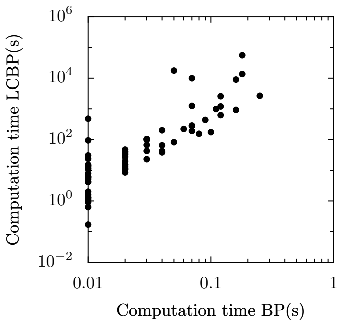


Figure 14: Computation time (in seconds) for PROMEDAS instances: (left) BP computation time vs. N; (center) LCBP computation time vs. N; (right) LCBP vs. BP.

LCBP, on the other hand, is the only method that gives a significant improvement over BP, in each case. Considering all patient cases, LCBP corrects the BP error with more than one order of magnitude in half of the cases for which BP was not already exact. The improvement obtained by LCBP has its price: the computation time of LCBP is rather large compared to that of BP, as shown in Figure 14. In many cases, this is due to a few rather large cavities. The cumulant-based loop correction methods are not available, due to the presence of factors involving more than two variables.

4. Discussion and Conclusion

We have proposed a method to improve the quality of the single-variable marginals calculated by an approximate inference method (e.g., BP) by correcting for the influence of loops in the factor graph. We have proved that the method is a generalization of BP if the initial approximate cavity distributions factorize and the factor graph does not contain short loops of exactly four nodes. If the factor graph does contain such short loops, we observe in many cases that the method reduces to the minimal CVM approximation if one applies it on factorized initial approximate cavity distributions. If, on the other hand, the LC method is applied in combination with BP estimates of the effective cavity interactions, we have seen that the loop-corrected error is approximately the square of the uncorrected BP error. Similar observations have been made for loop-corrected MF and TreeEP. For practical purposes, we suggest to apply loop corrections to BP ("LCBP"), because the loop correction approach requires many runs of the approximate inference method and BP is well suited for this job because of its speed. We have compared the performance of LCBP with other approximate inference methods that (partially) correct for the presence of loops. In most cases, LCBP turned out to be the most accurate method (with the notable exception of LCTreeEP, which is also considerably more expensive). LCBP still works for relatively strong interactions, in contrast with LCBP-Cum and LCBP-Cum-Lin.

On sparse factor graphs, TreeEP can obtain significant improvements over BP by correcting for loops that consist of part of the base tree and one additional interaction, using little computation time. However, for denser graphs, we observed that the difference between the quality of TreeEP and BP marginals diminishes. For both sparse and dense graphs, LCBP obtained more accurate results than TreeEP, although the computation time quickly increases for denser graphs.

We have seen that the CVM-Loop approximation, which uses small loops as outer clusters, can also provide accurate results, provided that the number of short loops is not too large and the number of intersections of clusters is limited. However, the computation time becomes prohibitive in many cases. In order to obtain the same accuracy as LCBP, the CVM-Loop approach usually needs significantly more computation time. This behavior is also seen on "real world" instances such as the ALARM network and PROMEDAS test cases. There may exist other cluster choices that give better results for the CVM approximation, but no general method for obtaining "good" cluster choices seems to be known (although for some special cases, for example, 2D grids, very good choices exist). Welling et al. (2005) give some criteria for "good" CVM cluster choices, but to our knowledge, no good general method for choosing CVM clusters is known.[11]

We have also compared the performance of LCBP with the original implementations proposed by Montanari and Rizzo (2005) (LCBP-Cum and LCBP-Cum-Lin) on the limited class of binary pairwise models. The original implementations work with cumulants instead of interactions and we believe that this explains the observed convergence difficulties of LCBP-Cum and LCBP-Cum-Lin in the regime of strong interactions. On sparse graphs, LCBP obtained better accuracy than LCBP-Cum and LCBP-Cum-Lin, using approximately similar computation time. This is mainly due to the fact that LCBP estimates the higher-order effective interactions in the cavity distributions. On dense graphs, both LCBP and LCBP-Cum become computationally infeasible. The linearized version LCBP-Cum-Lin, which is still applicable in these cases, performed surprisingly well, often obtaining similar accuracy as LCBP-Cum. For random graphs with high degree d (i.e., large Markov blankets), it turned out to be the most accurate of the applicable approximate inference methods. It is rather fortunate that the negative effect of the linearization error on the accuracy of the result becomes smaller as the degree increases, since it is precisely for high degree where one needs the linearization because of performance issues.

In the experiments reported here, the standard JunctionTree method was almost always faster than LCBP. The reason is that we have intentionally selected experiments for which exact inference is still feasible, in order to be able to compare the quality of various approximate inference methods. However, as implied by Figure 10, there is no reason to expect that LCBP will suddenly give inaccurate results when exact inference is no longer feasible. Thus we suggest that LCBP may be used to obtain accurate marginal estimates in cases where exact inference is impossible because of high treewidth. As illustrated in Figure 10, the computation time of LCBP scales very different from that of the JunctionTree method: whereas the latter is exponential in treewidth, LCBP is exponential in the size of the Markov blankets.

The fact that computation time of LCBP (in its current form) scales exponentially with the size of the Markov blankets can be a severe limitation in practice. Many real world Bayesian networks have large Markov blankets, prohibiting application of LCBP. The linear cumulant-based implementation LCBP-Cum-Lin does not suffer from this problem, as it is quadratic in the size of the Markov blankets. Unfortunately, this particular implementation can in its current form only be applied to graphical models that consist of binary variables and factors that involve at most two variables (which excludes any interesting Bayesian network, for example). Furthermore, problems may arise if some factors contain zeroes. For general application of loop correction methods, it will be of paramount importance to derive an implementation that combines the generality of LCBP

11. After submitting this manuscript, we became aware of the method called IJGP(i) proposed in Dechter et al. (2002). IJGP(i) is essentially a heuristic to create region graphs that can also significantly improve on BP. We have not yet done an experimental comparison of LCBP with IJGP(i).

with the speed of LCBP-Cum-Lin. This topic will be left for future research. The work presented here provides some intuition that may be helpful for constructing a general and fast loop correction method that is applicable to arbitrary factor graphs that can have large Markov blankets.

Another important direction for future research would be to find an extension of the loop correction framework that also gives a loop-corrected approximation of the normalization constant Z in (1). Additionally, and possibly related to that, it would be desirable to find an approximate "free energy", a function of the beliefs, whose stationary points coincide with the fixed points of Algorithm 1. This can be done for many approximate inference methods (MF, BP, CVM, EP) so it is natural to expect that the LC algorithm can also be seen as a minimization procedure of a certain approximate free energy. Despite some efforts, we have not yet been able to find such a free energy.

Recently, other loop correction approaches (to the Bethe approximation) have been proposed in the statistical physics community (Parisi and Slanina, 2006; Chertkov and Chernyak, 2006b). In particular, Chertkov and Chernyak (2006b) have derived a series expansion of the *exact* normalizing constant Z in terms of the BP solution. The first term of the series is precisely the Bethe free energy evaluated at the BP fixed point. The number of terms in the series is finite, but can be very large, even larger than the number of total states of the graphical model. Each term is associated with a "generalized loop", which is a subgraph of the factor graph for which each node has at least connectivity two. By truncating the series, it is possible to obtain approximate solutions that improve on BP by taking into account a subset of all generalized loops (Gómez et al., forthcoming; Chertkov and Chernyak, 2006a). Summarizing, the approach to loop corrections by Chertkov and Chernyak (2006b) takes a subset of loops into account in an exact way, whereas the loop correction approach presented in this article takes all loops into account in an approximate way. More experiments should be done to compare both approaches.

Summarizing, we have proposed a method to correct approximate inference methods for the influence of loops in the factor graph. We have shown that it can obtain very accurate results, also on real world graphical models, outperforming existing approximate inference methods in terms of quality, robustness or applicability. We have shown that it can be applied to problems for which exact inference is infeasible. The rather large computation time required is an issue which deserves further consideration; it may be possible to use additional approximations on top of the loop correction framework that trade quality for computation time.

Acknowledgments

The research reported here is part of the Interactive Collaborative Information Systems (ICIS) project (supported by the Dutch Ministry of Economic Affairs, grant BSIK03024) and was also sponsored in part by the Dutch Technology Foundation (STW). We thank Bastian Wemmenhove for stimulating discussions and for providing the PROMEDAS test cases.

Appendix A. Original Approach by Montanari and Rizzo (2005)

For completeness, we describe the implementation based on cumulants as originally proposed by Montanari and Rizzo (2005). The approach can be applied in recursive fashion. Here we will only discuss the first recursion level.

Consider a graphical model which has only binary (± 1-valued) variables and factors that involve at most two variables. The corresponding probability distribution can be parameterized in terms of the local fields $\{\theta_i\}_{i\in\mathcal{V}}$ and the couplings $\{J_{ij} = J_{ji}\}_{i\in\mathcal{V}, j\in\partial i}$:

$$P(x) = \frac{1}{Z}\exp\left(\sum_{i\in\mathcal{V}}\theta_i x_i + \frac{1}{2}\sum_{i\in\mathcal{V}}\sum_{j\in\partial i} J_{ij}x_i x_j\right).$$

Let $i \in \mathcal{V}$ and consider the corresponding cavity network of i. For $\mathcal{A} \subseteq \partial i$, the cavity *moment* $\mathcal{M}_{\mathcal{A}}^{\setminus i}$ is defined as the following expectation value under the cavity distribution:

$$\mathcal{M}_{\mathcal{A}}^{\setminus i} := \frac{\sum\limits_{x_{\partial i}} Z^{\setminus i}(x_{\partial i})\prod\limits_{j\in\mathcal{A}} x_j}{\sum\limits_{x_{\partial i}} Z^{\setminus i}(x_{\partial i})},$$

where we will not explicitly distinguish between approximate and exact quantities, following the physicists' tradition.[12] The cavity *cumulants* (also called "connected correlations") $\mathcal{C}_{\mathcal{A}}^{\setminus i}$ are related to the moments in the following way:

$$\mathcal{M}_{\mathcal{A}}^{\setminus i} = \sum_{\mathcal{B}\in\mathrm{Part}(\mathcal{A})}\prod_{\mathcal{E}\in\mathcal{B}}\mathcal{C}_{\mathcal{E}}^{\setminus i}$$

where $\mathrm{Part}(\mathcal{A})$ is the set of partitions of $\mathcal{A}$.

We introduce some notation: we define for $\mathcal{A} \subseteq \partial i$:

$$t_{i\mathcal{A}} := \prod_{k\in\mathcal{A}} \tanh J_{ik}.$$

Further, for a set X, we denote the even subsets of X as $\mathcal{P}_+(X) := \{Y \subseteq X : |Y| \text{ is even}\}$ and the odd subsets of X as $\mathcal{P}_-(X) := \{Y \subseteq X : |Y| \text{ is odd}\}$.

Using standard algebraic manipulations, one can show that for $j \in \partial i$, the expectation value of x_j in the absence of the interaction $\psi_{ij} = \exp(J_{ij}x_i x_j)$ can be expressed in terms of cavity moments of i as follows:

$$\frac{\sum\limits_{\mathcal{A}\in\mathcal{P}_+(\partial i\setminus j)} t_{i\mathcal{A}}\mathcal{M}_{\mathcal{A}\cup j}^{\setminus i} + \tanh\theta_i \sum\limits_{\mathcal{A}\in\mathcal{P}_-(\partial i\setminus j)} t_{i\mathcal{A}}\mathcal{M}_{\mathcal{A}\cup j}^{\setminus i}}{\sum\limits_{\mathcal{A}\in\mathcal{P}_+(\partial i\setminus j)} t_{i\mathcal{A}}\mathcal{M}_{\mathcal{A}}^{\setminus i} + \tanh\theta_i \sum\limits_{\mathcal{A}\in\mathcal{P}_-(\partial i\setminus j)} t_{i\mathcal{A}}\mathcal{M}_{\mathcal{A}}^{\setminus i}}. \tag{10}$$

On the other hand, the same expectation value can also be expressed in terms of cavity moments of j as follows:

$$\frac{\tanh\theta_j \sum\limits_{\mathcal{A}\in\mathcal{P}_+(\partial j\setminus i)} t_{j\mathcal{B}}\mathcal{M}_{\mathcal{B}}^{\setminus j} + \sum\limits_{\mathcal{A}\in\mathcal{P}_-(\partial j\setminus i)} t_{j\mathcal{B}}\mathcal{M}_{\mathcal{B}}^{\setminus j}}{\sum\limits_{\mathcal{A}\in\mathcal{P}_+(\partial j\setminus i)} t_{j\mathcal{B}}\mathcal{M}_{\mathcal{B}}^{\setminus j} + \tanh\theta_j \sum\limits_{\mathcal{A}\in\mathcal{P}_-(\partial j\setminus i)} t_{j\mathcal{B}}\mathcal{M}_{\mathcal{B}}^{\setminus j}}. \tag{11}$$

The consistency equations are now given by equating (10) to (11) for all $i \in \mathcal{V}$, $j \in \partial i$.

12. In Montanari and Rizzo (2005), the notation $\tilde{C}_{\mathcal{A}}^{(i)}$ is used for the cavity moment $\mathcal{M}_{\mathcal{A}}^{\setminus i}$.

The expectation value of x_i (in the presence of all interactions) can be similarly expressed in terms of cavity moments of i:

$$M_i := \sum_{x_i=\pm 1} P(x_i) x_i = \frac{\tanh\theta_i \sum\limits_{\mathcal{A}\in\mathcal{P}_+(\partial i)} t_{i\mathcal{A}} \mathcal{M}_{\mathcal{A}}^{\setminus i} + \sum\limits_{\mathcal{A}\in\mathcal{P}_-(\partial i)} t_{i\mathcal{A}} \mathcal{M}_{\mathcal{A}}^{\setminus i}}{\sum\limits_{\mathcal{A}\in\mathcal{P}_+(\partial i)} t_{i\mathcal{A}} \mathcal{M}_{\mathcal{A}}^{\setminus i} + \tanh\theta_i \sum\limits_{\mathcal{A}\in\mathcal{P}_-(\partial i)} t_{i\mathcal{A}} \mathcal{M}_{\mathcal{A}}^{\setminus i}}. \tag{12}$$

A.1 Neglecting Higher-order Cumulants

Montanari and Rizzo proceed by neglecting cavity cumulants $\mathcal{C}_{\mathcal{A}}^{\setminus i}$ with $|\mathcal{A}| > 2$. Denote by $\mathrm{Part}_2(\mathcal{A})$ the set of all partitions of $\mathcal{A}$ into subsets which have cardinality 2 at most. Thus, neglecting higher-order cavity cumulants amounts to the following approximation:

$$\mathcal{M}_{\mathcal{A}}^{\setminus i} \approx \sum_{\mathcal{B}\in\mathrm{Part}_2(\mathcal{A})} \prod_{\mathcal{E}\in\mathcal{B}} \mathcal{C}_{\mathcal{E}}^{\setminus i}. \tag{13}$$

By some algebraic manipulations, one can express the consistency Equations (10) = (11) in this approximation as follows:

$$\mathcal{M}_j^{\setminus i} = \frac{\tanh\theta_j \sum\limits_{\mathcal{A}\in\mathcal{P}_+(\partial j\setminus i)} t_{j\mathcal{B}} \mathcal{M}_{\mathcal{B}}^{\setminus j} + \sum\limits_{\mathcal{A}\in\mathcal{P}_-(\partial j\setminus i)} t_{j\mathcal{B}} \mathcal{M}_{\mathcal{B}}^{\setminus j}}{\sum\limits_{\mathcal{A}\in\mathcal{P}_+(\partial j\setminus i)} t_{j\mathcal{B}} \mathcal{M}_{\mathcal{B}}^{\setminus j} + \tanh\theta_j \sum\limits_{\mathcal{A}\in\mathcal{P}_-(\partial j\setminus i)} t_{j\mathcal{B}} \mathcal{M}_{\mathcal{B}}^{\setminus j}}$$
$$- \sum_{k\in\partial i\setminus j} t_{ik} \mathcal{C}_{jk}^{\setminus i} \frac{\tanh\theta_i \sum\limits_{\mathcal{A}\in\mathcal{P}_+(\partial i\setminus\{j,k\})} t_{i\mathcal{A}} \mathcal{M}_{\mathcal{A}}^{\setminus i} + \sum\limits_{\mathcal{A}\in\mathcal{P}_-(\partial i\setminus\{j,k\})} t_{i\mathcal{A}} \mathcal{M}_{\mathcal{A}}^{\setminus i}}{\sum\limits_{\mathcal{A}\in\mathcal{P}_+(\partial i\setminus j)} t_{i\mathcal{A}} \mathcal{M}_{\mathcal{A}}^{\setminus i} + \tanh\theta_i \sum\limits_{\mathcal{A}\in\mathcal{P}_-(\partial i\setminus j)} t_{i\mathcal{A}} \mathcal{M}_{\mathcal{A}}^{\setminus i}}. \tag{14}$$

One can use (13) to write (14) in terms of the singleton cumulants $\{\mathcal{M}_j^{\setminus i}\}_{i\in\mathcal{V},j\in\partial i}$ and the pair cumulants $\{\mathcal{C}_{jk}^{\setminus i}\}_{i\in\mathcal{V},j\in\partial i,k\in\partial i\setminus j}$. Given (estimates of) the pair cumulants, the consistency Equations (14) are thus fixed point equations in the singleton cumulants. The procedure is now:

- Estimate the pair cumulants $\{\mathcal{C}_{jk}^{\setminus i}\}_{i\in\mathcal{V},j\in\partial i,k\in\partial i\setminus j}$ using BP in combination with linear response (called “response propagation” in Montanari and Rizzo (2005)).
- Calculate the fixed point $\{\mathcal{M}_j^{\setminus i}\}_{i\in\mathcal{V},j\in\partial i}$ of (14) using the estimated pair cumulants.
- Use (12) in combination with (13) to calculate the final expectation values $\{M_j\}_{j\in\mathcal{V}}$ using the estimated pair cumulants and the fixed point of (14).

A.2 Linearized Version

The update equations can be linearized by expanding up to first order in the pair cumulants $\mathcal{C}_{jk}^{\setminus i}$. This yields the following linearized consistency equation (Montanari and Rizzo, 2005):

$$\mathcal{M}_j^{\setminus i} = T_i^{\setminus j} - \sum_{l\in\partial i\setminus j} \Omega_{j,l}^{\setminus i} t_{il} \mathcal{C}_{jl}^{\setminus i} + \sum_{\{l_1,l_2\}:l_1,l_2\in\partial j\setminus i} \Gamma_{i,l_1 l_2}^{\setminus j} t_{jl_1} t_{jl_2} \mathcal{C}_{l_1 l_2}^{\setminus j} \tag{15}$$

where

$$\begin{aligned}
T_{\mathcal{A}}^{\setminus i} &:= \tanh\left(\theta_i + \sum_{k\in\partial i\setminus\mathcal{A}} \tanh^{-1}\left(t_{ik}\mathcal{M}_k^{\setminus i}\right)\right), \\
\Omega_{j,l}^{\setminus i} &:= \frac{T_{jl}^{\setminus i}}{1 + t_{il}M_l^{\setminus i}T_{jl}^{\setminus i}}, \\
\Gamma_{i,l_1l_2}^{\setminus j} &:= \frac{T_{il_1l_2}^{\setminus j} - T_i^{\setminus j}}{1 + t_{jl_1}t_{jl_2}\mathcal{M}_{l_1}^{\setminus j}\mathcal{M}_{l_2}^{\setminus j} + t_{jl_1}\mathcal{M}_{l_1}^{\setminus j}T_{il_1l_2}^{\setminus j} + t_{jl_2}\mathcal{M}_{l_2}^{\setminus j}T_{il_1l_2}^{\setminus j}}.
\end{aligned}$$

The final magnetizations (12) are, up to first order in the pair cumulants:

$$M_j = T^{\setminus j} + \sum_{\{l_1,l_2\}:l_1,l_2\in\partial j^2} \Gamma_{l_1l_2}^{\setminus j} t_{jl_1}t_{jl_2}\mathcal{C}_{l_1l_2}^{\setminus j} + O(\mathcal{C}^2)$$

where

$$\Gamma_{l_1l_2}^{\setminus j} := \frac{T_{l_1l_2}^{\setminus j} - T^{\setminus j}}{1 + t_{jl_1}t_{jl_2}M_{l_1}^{\setminus j}\mathcal{M}_{l_2}^{\setminus j} + t_{jl_1}\mathcal{M}_{l_1}^{\setminus j}T_{l_1l_2}^{\setminus j} + t_{jl_2}\mathcal{M}_{l_2}^{\setminus j}T_{l_1l_2}^{\setminus j}}.$$

References

H. Bethe. Statistical theory of superlattices. *Proc. R. Soc. A*, 150:552–575, 1935.

M. Chertkov and V. Y. Chernyak. Loop calculus helps to improve belief propagation and linear programming decodings of low-density-parity-check codes. *arXiv.org preprint*, arXiv:cs/0609154v1 [cs.IT], 2006a. URL `http://arxiv.org/abs/cs/0609154v1`.

M. Chertkov and V. Y. Chernyak. Loop series for discrete statistical models on graphs. *Journal of Statistical Mechanics: Theory and Experiment*, 2006(06):P06009, 2006b. URL `http://stacks.iop.org/1742-5468/2006/P06009`.

R. Dechter, K. Kask, and R. Mateescu. Iterative join-graph propagation. In *Proceedings of the 18th Annual Conference on Uncertainty in Artificial Intelligence (UAI-02)*, pages 128–13, San Francisco, CA, 2002. Morgan Kaufmann.

G. Elidan, I. McGraw, and D. Koller. Residual belief propagation: Informed scheduling for asynchronous message passing. In *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, Boston, Massachussetts, July 2006.

V. Gómez, J. M. Mooij, and H. J. Kappen. Truncating the loop series expansion for belief propagation. *Journal of Machine Learning Research*, forthcoming. URL `http://arxiv.org/abs/cs/0612030v1`.

T. Heskes, C. A. Albers, and H. J. Kappen. Approximate inference and constrained optimization. In *Proceedings of the 19th Annual Conference on Uncertainty in Artificial Intelligence (UAI-03)*, pages 313–320, San Francisco, CA, 2003. Morgan Kaufmann Publishers.

T. Jaakkola and M. I. Jordan. Variational probabilistic inference and the QMR-DT network. *Journal of Artificial Intelligence Research*, 10:291–322, 1999. URL `http://www.jair.org/papers/paper583.html`.

R. Kikuchi. A theory of cooperative phenomena. *Phys. Rev.*, 81:988–1003, 1951.

F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Inform. Theory*, 47(2):498–519, February 2001.

M. Mézard, G. Parisi, and M. A. Virasoro. *Spin Glass Theory and Beyond*. World Scientific, Singapore, 1987.

T. Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 362–369, San Francisco, CA, 2001. Morgan Kaufmann Publishers.

T. Minka and Y. Qi. Tree-structured approximations by expectation propagation. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press.

A. Montanari and T. Rizzo. How to compute loop corrections to the Bethe approximation. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(10):P10011, 2005. URL `http://stacks.iop.org/1742-5468/2005/P10011`.

J. M. Mooij and H. J. Kappen. On the properties of the Bethe approximation and loopy belief propagation on binary networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2005 (11):P11012, 2005. URL `http://stacks.iop.org/1742-5468/2005/P11012`.

J. M. Mooij and H. J. Kappen. Loop corrections for approximate inference. *arXiv.org preprint*, arXiv:cs/0612030v1 [cs.AI], 2006. URL `http://arxiv.org/abs/cs/0612030v1`.

J. M. Mooij, B. Wemmenhove, H. J. Kappen, and T. Rizzo. Loop corrected belief propagation. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS-07)*, 2007.

G. Parisi. *Statistical Field Theory*. Addison-Wesley, Redwood City, Ca, 1988.

G. Parisi and F. Slanina. Loop expansion around the Bethe-Peierls approximation for lattice models. *Journal of Statistical Mechanics: Theory and Experiment*, 2006(02):L02003, 2006. URL `http://stacks.iop.org/1742-5468/2006/L02003`.

J. Pearl. *Probabilistic Reasoning in Intelligent systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, CA, 1988.

A. Pelizzola. Cluster variation method in statistical physics and probabilistic graphical models. *J. Phys. A: Math. Gen.*, 38:R309–R339, August 2005.

M. A. Shwe, B. Middleton, D. E. Heckerman, M. Henrion, E. J. Horvitz, H. P. Lehmann, and G. F. Cooper. Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base. I. The probabilistic model and inference algorithms. *Methods of information in Medicine*, 30(4):241–255, October 1991.

M. Takikawa and B. D'Ambrosio. Multiplicative factorization of noisy-max. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 622–63, San Francisco, CA, 1999. Morgan Kaufmann.

M. Welling, T. Minka, and Y. W. Teh. Structured region graphs: Morphing EP into GBP. In *Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence (UAI-05)*, page 609, Arlington, Virginia, 2005. AUAI Press.

W. Wiegerinck, H. J. Kappen, E. W. M. T. ter Braak, W. J. P. P. ter Burg, M. J. Nijman, Y. L. O, and J. P. Neijt. Approximate inference for medical diagnosis. *Pattern Recognition Letters*, 20: 1231–1239, 1999.

J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51(7):2282–2312, July 2005.

A. L. Yuille. CCCP algorithms to minimize the Bethe and Kikuchi free energies: convergent alternatives to belief propagation. *Neural Computation*, 14(7):1691–1722, 2002.

Journal of Machine Learning Research 8 (2007) 1145-1164 Submitted 11/05; Revised 1/07; Published 5/07

Penalized Model-Based Clustering with Application to Variable Selection

Wei Pan WEIP@BIOSTAT.UMN.EDU
Division of Biostatistics
School of Public Health
University of Minnesota
Minneapolis, MN 55455, USA

Xiaotong Shen XSHEN@STAT.UMN.EDU
School of Statistics
University of Minnesota
Minneapolis, MN 55455, USA

Editor: Bin Yu

Abstract

Variable selection in clustering analysis is both challenging and important. In the context of model-based clustering analysis with a common diagonal covariance matrix, which is especially suitable for "high dimension, low sample size" settings, we propose a penalized likelihood approach with an L_1 penalty function, automatically realizing variable selection via thresholding and delivering a sparse solution. We derive an EM algorithm to fit our proposed model, and propose a modified BIC as a model selection criterion to choose the number of components and the penalization parameter. A simulation study and an application to gene function prediction with gene expression profiles demonstrate the utility of our method.

Keywords: BIC, EM, mixture model, penalized likelihood, soft-thresholding, shrinkage

1. Introduction

This article concerns variable selection in model-based clustering, especially for "high dimension, low sample size" data, where the data dimension greatly exceeds the number of observations. Specifically, given n P-dimensional observations $x_j = (x_{j1}, ..., x_{jP})'$ for $j = 1, ..., n$, we aim to group the data into a few, say K, clusters such that the observations in the same cluster are more similar to each other than those from different clusters. In this context, some of the attributes x_{jp}'s of x_j may not be relevant: use of such attributes only introduces noise, and may impede uncovering the clustering structure of interest. In addition, removing non-informative attributes may largely enhance interpretability. Due to lack of statistical models in many existing clustering algorithms, it is difficult to implement principled variable selection, though some promising methods have been proposed, based largely on heuristics (Friedman and Meulman, 2004; Mangasarian and Wild, 2004). In contrast, model-based clustering (McLachlan and Peel, 2002; Fraley and Raftery, 2002) assumes that data come from a finite mixture model with each component corresponding to a cluster; with such a statistical model, statistical inference, including variable selection, can be carried out. Because, to our knowledge, no formal hypothesis tests are available to assess the statistical significance of an attribute, it is unclear how to implement a sequential variable selection, such as forward additions and/or backward eliminations of variables, in model-based clustering; an alternative is to conduct best subset selection, which however is unrealistic for high-dimensional data: for example, with $P = 1000$, there are more than 10^{300} possible models to be considered, which is prohibitive given the current standard computing power. Furthermore, even for smaller problems, as in regression, due to its discreteness, best subset selection may be unstable and may not work well in selecting relevant variables (Tibshirani, 1996); most importantly, unlike in regression or classification but unique to clustering or semi-supervised learning, best subset selection may identify a correct model which however is of no interest, as to be confirmed by our numerical example later.

With high-dimensional data, as an alternative to variable selection, one may apply dimension reduction techniques, such as principal component analysis, prior to clustering (Ghosh and Chinnaiyan, 2002; Liu et al., 2003). A possible drawback of this approach is the separation between dimension reduction and subsequent clustering; for example, as pointed out by many researchers (Chang, 1983; Yeung and Ruzzo, 2001; Raftery, 2003), using first few principal components in clustering may destroy the clustering structure of the original data.

There has been increasing interest in variable selection for model-based clustering, mostly within the Bayesian framework (Liu et al., 2003; Hoff, 2005, 2006; Tadesse et al., 2005; Raftery and Dean, 2006; Kim et al., 2006). An idea is to parametrize the mean of cluster k as $\mu_k = \mu + \delta_k$, where μ is the global mean. It is clear that, if some components of δ_k are 0, then the corresponding attributes are not informative to clustering, at least in terms of the means/locations. Two Bayesian approaches have been proposed based on this idea (Liu et al., 2003; Hoff, 2005, 2006). Another Bayesian approach, analogous to stepwise variable selection in regression, is to sequentially compare two nested models to determine whether an attribute should be included in or excluded from the current model based on a greedy search (Raftery and Dean, 2006), which may be computationally too time-consuming for high-dimensional data. In contrast, to our knowledge, no frequentist alternatives to subset selection are available for variable selection in model-based clustering. In light of the success of penalized regression with variable selection (Tibshirani, 1996; Fan and Li, 2001), we conjecture that penalization may be also viable to variable selection in clustering, and hence we propose an approach through penalized model-based clustering. Specifically, cluster-specific means μ_k are adaptively shrunken towards the global mean μ; with an appropriately chosen penalty function, some components of μ_k are estimated to be exactly the same as that of μ, effectively realizing variable selection. We also propose a modified BIC as a model selection criterion to adaptively determine the amount of penalization as well as the number of clusters. Note that, although there is an extensive body of literature on penalized likelihood methods, most focus on classification and regression; in particular, to our knowledge, we are not aware of any existing works on penalized likelihood particularly designed for multivariate clustering.

Recent advances in high-throughput biotechnologies, such as microarrays, have generated a large amount of high-dimensional data, and have led to routine use of clustering analyses, for example, in gene function discovery (Eisen et al., 1998) and cancer subtype discovery (Golub et al., 1999; Ghosh and Chinnaiyan, 2002; McLachlan et al., 2002; Yeung et al., 2001). In these applications, one key issue is how to select variables: although expression levels of thousands or tens of thousands of genes are measured on each microarray, corresponding to a high-dimensional observation, it is known that not all the genes are related to the phenotype of interest, for example, subtypes of a cancer; in fact, often only a small number of the genes are relevant, and identifying those genes is one of the biologically most important goals. Hence, variable selection in clustering not only improves the performance in identifying interesting clusters, as to be shown later, but also largely facilitates interpretation of results, and even directly addresses biological questions of interest, for example, which genes are involved in the biology of a cancer or its subtypes. In this article, in addition to the promising application of cancer subtype discovery, we also apply model-based clustering to the task of gene function discovery (Li and Hong, 2001; Ghosh and Chinnaiyan, 2002). Although the human genome and many other genome sequencing projects have led to a discovery of many new genes, biological functions of many genes remain unknown; many known functions also need to be refined. It has become popular to cluster gene expression profiles to discover unknown gene functions.

In the remaining parts of this article, we first review briefly the standard model-based clustering, then we introduce a general framework for penalized model-based clustering. We propose a specific implementation with an L_1 penalty, resulting in soft-thresholding on the mean parameters, and thus realizing automatic variable selection. We derive an EM algorithm to compute the maximum penalized likelihood estimates for the model; a modified BIC is used to determine the number of components and the value of the penalization parameter in penalized model-based clustering. We compare the proposed method with the standard method using simulated data and gene expression data for tumor subtype discovery and gene function prediction; in particular, we illustrate problems associated with clustering without variable selection, and those with best subset selection, concluding that penalized clustering is an effective and simple method for variable selection. We end the article with a short discussion on some open questions.

2. Methods

We first give a brief review on model-based clustering with a finite Normal mixture model, then we introduce our penalized model-based clustering, including an EM algorithm and a modified BIC for model selection.

2.1 Model-based Clustering

In model-based clustering, it is assumed that each observation x is drawn from a finite mixture distribution $f(x;\Theta) = \sum_{k=1}^{K} \pi_k f_k(x;\theta_k)$, with the mixing proportion π_k, component-specific distribution f_k and its parameters θ_k. Denote by $\Theta = \{(\pi_k, \theta_k) : k = 1, ..., K\}$ all unknown parameters, with restriction that $0 \le \pi_k \le 1$ for any k and $\sum_{k=1}^{K} \pi_k = 1$. Each component of the mixture distribution corresponds to a cluster. The number of clusters, K, has to be determined in practice; see section 2.4.

Given data x_j, $j = 1, ..., n$, the log-likelihood is

$$\log L(\Theta) = \sum_{j=1}^{n} \log[\sum_{k=1}^{K} \pi_k f_k(x_j;\theta_k)].$$

Maximization of the above log-likelihood with respect to Θ is difficult, and it is common to use the EM algorithm (Dempster et al., 1977) by casting the problem in the framework of missing data. Define z_{kj} as the indicator of whether x_j is from component k; that is, $z_{kj} = 1$ if x_j is indeed from component k, and $z_{kj} = 0$ otherwise. If the missing data z_{kj}'s could be observed, then the log-likelihood for the complete data is:

$$\log L_c(\Theta) = \sum_k \sum_j z_{kj} [\log \pi_k + \log f_k(x_j;\theta_k)].$$

The EM algorithm can be applied to obtain the maximum likelihood estimator (MLE) of Θ; see McLachlan and Peel (2002) and Fraley and Raftery (2002) for more details.

2.2 Penalized Model-based Clustering

With the same motivation as in penalized regression, we propose a penalized model-based clustering approach. The general purpose of penalization is for model regularization, which in general can enhance the predictive power of a model, and may be even necessary in some situations. For example, in the univariate Normal mixture model with each $f_k(.) = \phi(.;\mu_k, \sigma_k)$, it is well-known that with $\sigma_k \to 0$, we have a degeneracy with $\log L$ being unbounded, and thus no (unrestricted) MLE exists. Ciuperca et al. (2003) proposed a penalized likelihood approach to dealing with the degeneracy: by penalizing small variance components σ_k, one circumvents the problem. In addition, as to be discussed in the next section, with an appropriate choice of penalty function, we can realize a sparse solution, resulting in automatic variable selection.

Specifically, we regularize $\log L(\Theta)$ to yield a penalized log-likelihood:

$$\log L_P(\Theta) = \sum_{j=1}^{n} \log[\sum_{k=1}^{K} \pi_k f_k(x_j;\theta_k)] - h_\lambda(\Theta),$$

where $h_\lambda()$ is a penalty function with penalization parameter λ. The choice of $h_\lambda()$ depends on the goal of the analysis; see Fan and Li (2001) for some general theory. Correspondingly, the penalized log-likelihood for the complete data is

$$\log L_{c,P}(\Theta) = \sum_{k=1}^{K} \sum_{j=1}^{n} z_{kj}[\log \pi_k + \log f_k(x_j;\theta_k)] - h_\lambda(\Theta).$$

We propose using such penalized model-based clustering as a general way to regularize parameter estimates. This can be useful for high-dimensional data, especially for situations of "large P, small n". Recently Fraley and Raftery (2005) proposed a Bayesian approach to regularizing model-based clustering; there is a large body of literature on Bayesian mixture modeling, for example, Richardson and Green (1997), Jasra et al. (2005) and references therein. There is a well known connection between penalized likelihood and Bayesian modeling (Hastie et al., 2001): it can be regarded that minus the penalty function $-h_\lambda(\Theta)$ is proportional to the log density of the prior distribution for parameters Θ, and the penalized (log) likelihood is proportional to the (log) posterior density.

Note that in contrast to Ciuperca et al. (2003), where only univariate Normal mixture models were considered, our main interest here is in multivariate clustering.

2.3 Penalizing Mean Parameters

Now we propose a specific implementation of penalized model-based clustering to realize variable selection. Consider the common case with each component f_k as Normal. We are particularly interested in "large P, small n" often encountered in genomic studies. Hence, as in naive Bayes classification, we adopt a working independence model for components of x_j. Furthermore, to facilitate variable selection for "large P, small n" settings, a common diagonal covariance matrix is used across clusters; more discussions on this choice is given in Section 4. We assume throughout this article that, prior to clustering analysis, we have standardized data so that each attribute has sample mean 0 and sample variance 1. Specifically, we have

$$f_k(x;\theta_k) = \frac{1}{(2\pi)^{P/2}|V|}\exp\left(-\frac{1}{2}(x-\mu_k)'V^{-1}(x-\mu_k)\right),$$

where $V = diag(\sigma_1, \sigma_2, ..., \sigma_P)$, and $|V| = \prod_{p=1}^{P}\sigma_p$. We propose using the L_1 penalty:

$$h_\lambda(\Theta) = \lambda\sum_k\sum_p|\mu_{kp}|,$$

though other penalty functions may be also suitable, as discussed in Fan and Li (2001) in the context of regression. The main goal is to obtain a sparse solution with many small estimates of μ_{kp}'s automatically set to 0, thus realizing variable selection.

Next we derive an EM algorithm for the above penalized model-based clustering; in particular, it is confirmed that the L_1-penalty yields a thresholding rule with the desired sparsity property. The derivation closely follows from that for standard model-based clustering (McLachlan and Peel, 2002) and the general methodology for penalized likelihood (Green, 1990). We use generic notation $\Theta^{(m)}$ to represent the parameter estimates at iteration m, and use $X = (x_1, ..., x_n)$ to denote all the observations. It is easy to verify that the E-step yields

$$Q_P(\Theta;\Theta^{(m)}) = E_{\Theta^{(m)}}(\log L_{c,P}|X) = \sum_k\sum_j\tau_{kj}^{(m)}[\log\pi_k + \log f_k(x_j;\theta_k)] - \lambda\sum_k\sum_p|\mu_{kp}|,$$

where

$$\tau_{kj}^{(m)} = \frac{\pi_k^{(m)}f_k(x_j;\theta_k^{(m)})}{f(x_j;\Theta^{(m)})} = \frac{\pi_k^{(m)}f_k(x_j;\theta_k^{(m)})}{\sum_{k=1}^{K}\pi_k^{(m)}f_k(x_j;\theta_k^{(m)})} \tag{1}$$

is the estimated posterior probability of x_j's coming from component k.

The M-step maximizes the above Q_P to update the parameter estimates. It is easy to show that

$$\frac{\partial Q_P}{\partial\pi_k} = \sum_j(\tau_{kj}^{(m)}/\pi_k - \tau_{Kj}^{(m)}/\pi_K),$$

for any $k = 1, 2, ..., K-1$, and

$$\frac{\partial Q_P}{\partial\sigma_p^2} = \sum_k\sum_j\tau_{kj}^{(m)}\left[-\frac{1}{2\sigma_p^2} + \frac{(x_{jp}-\mu_{kp})^2}{2\sigma_p^4}\right],$$

for any $p = 1, ..., P$. Hence

$$\hat{\pi}_k^{(m+1)} = \sum_{j=1}^{n}\tau_{kj}^{(m)}/n, \quad \text{and} \quad \hat{\sigma}_p^{2,(m+1)} = \sum_{k=1}^{K}\sum_{j=1}^{n}\tau_{kj}^{(m)}(x_{jp}-\mu_{kp}^{(m)})^2/n. \tag{2}$$

Now for the mean parameters,

$$\frac{\partial Q_P}{\partial\mu_k} = \sum_j\tau_{kj}^{(m)}V^{-1}(x_j-\mu_k) - \lambda\text{sign}(\mu_k).$$

Some algebraic manipulations yield

$$\hat{\mu}_k^{(m+1)} = \text{sign}(\tilde{\mu}_k^{(m+1)}) \left(|\tilde{\mu}_k^{(m+1)}| - \frac{\lambda}{\sum_j \tau_{kj}^{(m+1)}} V^{(m+1)} 1 \right)_+ , \tag{3}$$

where $\tilde{\mu}_k^{(m+1)} = \sum_j \tau_{kj}^{(m+1)} x_j / \sum_j \tau_{kj}^{(m+1)}$ is the usual update for μ_k if no penalty is imposed; for any f, $f_+ = f$ if $f > 0$, and $f_+ = 0$ otherwise; 1 is a vector with all elements 1's. Note that all the operations in (3), including sign() and $()_+$, are component-wise. It is evident that, if $\lambda > |\sum_{j=1}^n \tau_{kj}^{(m+1)} x_{jp} / \sigma_p^{2,(m+1)}|$, then $\hat{\mu}_{kp}^{(m+1)} = 0$; otherwise, $\hat{\mu}_{kp}^{(m+1)}$ is obtained by shrinking $\tilde{\mu}_{kp}^{(m+1)}$ towards 0 by an amount $\lambda \sigma_p^{2,(m+1)} / \sum_{j=1}^n \tau_{kj}^{(m+1)}$.

The above iteration is repeated until convergence, resulting in the maximum penalized likelihood estimate (MPLE) $\hat{\Theta}$. Then we use (1) to calculate the posterior probability of any observation x's belonging to each cluster, and assign the observation to the cluster with the highest probability. Because of possible existence of multiple local maxima, we run the algorithm multiple times, and each time we use the result from a randomly started K-means algorithm as starting values for the EM. We fit a series of models with various values of K and λ, then use a model selection criterion to choose their appropriate values, as to be discussed in the next section.

It can be seen that, if $\hat{\mu}_{kp} = 0$ for all k, then the p-th attribute does not contribute to clustering: it will be cancelled out from the numerator and the denominator of (1). In contrast, in the standard method, all attributes contribute to the posterior probability calculation.

Note that in (3), if we use $\tilde{\mu}_k^{(m+1)}$, instead of $\hat{\mu}_k^{(m+1)}$, we obtain the standard model-based clustering, which is equivalent to using $\lambda = 0$. In our numerical examples, to reduce bias, we used $\tilde{\mu}_{kp}^{(m)}$ in (2) to estimate σ_p^2, though we did not find much difference in several simulations if $\hat{\mu}_{kp}^{(m)}$ was indeed used. In addition, if we replace $\hat{\mu}_k^{(m+1)}$ by

$$\hat{\mu}_{k,H}^{(m+1)} = \tilde{\mu}_k^{(m+1)} I(\lambda > |\sum_{j=1}^n \tau_{kj}^{(m+1)} V^{-1,(m)} x_j|),$$

we obtain so-called hard-thresholding, which is in contrast to soft-thresholding in (3). In our numerical examples, we found that hard-thresholding gave results similar to those of soft-thresholding, and we will skip its further discussion.

2.4 Model Selection

In practice, we need to determine the number of components, K. This is realized by first fitting a series of models with various numbers of components, and then using a model selection criterion to choose the best one. For standard model-based clustering, it is common to use Bayesian information criterion (BIC) (Schwarz, 1978) defined as

$$BIC = -2 \log L(\tilde{\Theta}) + \log(n) d,$$

where $\tilde{\Theta}$ is the MLE, and $d = \dim(\Theta)$ is the total number of unknown parameters (Fraley and Raftery, 1998). In our proposed model, we have $d = K + P + KP - 1$, because we have three sets of parameters, π_k's, σ_p's and μ_{kp}'s, under the constraint $\sum_{k=1}^K \pi_k = 1$.

For penalized model-based clustering, in addition to K, we also have to choose an appropriate value of penalization parameter λ; a model selection criterion has to account for the adaptive choice of λ. One difficulty in using the above BIC criterion is that it is not always clear what is d in a penalized model. Although other resampling-based model selection methods, such as cross-validation or generalized degrees of freedom (Shen and Ye, 2002) can be employed, they are computationally more demanding, and even prohibitive for large and/or high-dimensional data as considered here. Following a conjecture of Efron et al. (2004) and a result of Zou et al. (2004) for L_1-penalized regression, we treat d as the number of non-zero parameter estimates, modifying BIC for penalized model-based clustering as

$$BIC = -2\log L(\hat{\Theta}) + \log(n)d_e,$$

where $\hat{\Theta}$ is the MPLE, and $d_e = K + P + KP - 1 - q$ is the effective number of parameters; we set q as the number of the MPLE mean components that equal to 0. Hence, as expected, due to thresholding, $d_e < d$ with a large penalization parameter λ.

3. Results

We first present results for simulated data, then consider clustering samples and clustering genes for two microarray data.

3.1 Simulated Data

We consider first high-dimensional data, then, to facilitate comparisons with best subset selection, we also consider low-dimensional data.

3.1.1 LARGE P

We first considered simulated data as described in Hoff (2004) and Hoff (2005). In each simulated data set, there were two clusters based on the first 150 attributes, while only one cluster based on the remaining 850 attributes; in other words, there were a total of $P = 1000$ variables with the first 150 effective while the other 850 as noise variables in forming two clusters. Specifically, there were $n = 100$ observations with 85 in one cluster and 15 in the other: the first 150 variables were iid from $N(0,1)$ for the first cluster, whereas they were iid from $N(1.5,1)$ for the second cluster; the remaining 850 variables were all iid from $N(0,1)$ for either cluster. Hence, there were 150 informative attributes and 850 noise ones.

For each of 100 simulated data sets, we fitted a series of models with the number of components $K = 1,2,3$ and various values of penalization parameter $\lambda = 0, 1, 1.5, 2, 5, 7.5, 10, 12.5, 15, 17.5, 20, 25$ and 30.

Table 1 summarizes the means and standard errors of BIC for the standard clustering using all 1000 attributes, and those of BIC and penalization parameter λ of the selected models in penalized clustering. Table 2 gives the frequencies of the selected K for both the methods. Twenty out of a hundred times, standard model-based clustering incorrectly selected $K = 1$, failing to discover the existence of the two clusters of interest. In some sense, the result was reasonable and unsurprising: indeed, there were two clusters based on the first 150 attributes in the data; however, based on any of the other 850 attributes, there was only one cluster. Because standard clustering used all the attributes, noting that 850 was much larger than 150, as expected it might choose $K = 1$. In contrast, with an appropriate variable selection, penalized clustering more frequently chose the model with

	Standard	Penalized	
K	BIC	BIC	λ
1	92923 (2)	88393 (0)	1.00 (0.00)
2	92834 (12)	85738 (65)	9.60 (0.22)
3	96282 (13)	86778 (32)	9.60 (0.12)

Table 1: Mean BIC (with standard errors in parentheses) with various numbers (K) of clusters in the standard and penalized clustering for 100 simulated data sets. λ is the value of the penalization parameter minimizing BIC for the given K.

	Standard		Penalized				
K	Freq	BIC	Freq	BIC	λ	#Zero1	#Zero0
1	20	92923	0	-	-	-	-
		(5)					
2	80	92791	94	85679	9.57	1.1	832.5
		(10)		(64)	(0.22)	(0.2)	(1.7)
3	0	-	6	86348	7.50	0.0	626.5
				(43)	(0.00)	(0.0)	(5.6)

Table 2: Frequencies of the selected numbers (K) of clusters in the standard and penalized clustering from 100 simulated data sets with $P = 1000$. The corresponding means (with standard errors in parentheses) of BIC, λ, the number of the first 150 informative attributes excluded (#Zero1), and the number of the last 850 noise attributes excluded (#Zero0) are also included.

	Standard	Penalized				
K	Freq	Freq	#(1 and 2)	#(1 or 2, not both)	#Zero1	#Zero0
1	100	6	0	0	2	8
					(0)	(0)
2	0	38	30	6	0.26	5.08
					(0.09)	(0.26)
3	0	56	42	12	0.29	2.45
					(0.07)	(0.19)

Table 3: Frequencies of the selected numbers (K) of clusters in the standard and penalized clustering from 100 simulated data sets with $P = 10$. The frequencies of the corresponding models including both the two informative attributes (#(1 and 2)), or only one of the two (#(1 or 2, not both)), and the means (with standard errors in parentheses) of the number of the two informative attributes excluded (#Zero1), and the number of the other 8 noise attributes excluded (#Zero0) are also included.

$K = 2$, uncovering the interesting structure in the data. Importantly, the penalized approach can automatically select attributes: out of the total 850 noise attributes, on average, 833 attributes were correctly identified and not used in the final clustering; on the other hand, only one out of 150 informative attributes was not used.

Penalized clustering gave perfect assignments for $K = 2$: the 15 and 85 observations from two distributions/classes were correctly assigned to clusters 1 and 2 respectively. More interestingly, even when $K = 3$ was selected, the assignments were also correct; the clustering results for the 6 simulated data sets with $K = 3$ were the following: i) for two data sets, the 15 observations from class 1 were assigned to cluster 1, and 45 and 40 observations from class 2 were assigned to clusters 2 and 3 respectively, denoted as $\{(15,0,0),(0,45,40)\}$; ii) for two data sets: $\{(15,0,0),(0,49,36)\}$; iii) for the other two, $\{(15,0,0),(0,48,37)\}$ and $\{(15,0,0),(0,46,39)\}$ respectively.

It would be interesting to compare our method with best subset selection, which however was computationally prohibitive: with 1000 attributes, there were $2^{1000} \approx 10^{300}$ possible subsets/models! Note that, because there is no formal significance test for each individual attribute in model-based clustering, the commonly used sequential variable selection in regression is not applicable here. Below, we considered a problem with a much smaller P so that a comparison with best subset selection was possible.

3.1.2 SMALL P

We considered simulated data similar to those in the previous section, but with much fewer attributes. There were only $P = 10$ attributes, among which the first two were informative while the other eight were noise attributes; all other aspects remained the same. In best subset selection, first, each of the 1023 (non-null) candidate models containing all possible combinations of the 10 attributes was fitted using `Mclust()` in R with K ranging from 1 to 3 (and a common diagonal covariance matrix); for each model, the value of K was selected to give the minimum BIC; finally, we chose the final model from the 1023 fitted models as the one with the smallest BIC.

	Any attributes			≥ 2 attributes		
K	Freq	#(1 and 2)	#(1 or 2, not both)	Freq	#(1 and 2)	#(1 or 2, not both)
1	79	0	0	56	11	6
2	19	0	7	41	21	11
3	2	0	1	3	2	0

Table 4: Frequencies of the selected numbers (K) of clusters by best subset selection from 100 simulated data sets with $P = 10$. The frequencies of the corresponding models including both the two informative attributes (#(1 and 2)), or only one of the two (#(1 or 2, not both)). Two searches were conducted: all possible models including any combinations of the attributes, and only models including at least two attributes.

Table 3 gives the results for standard and penalized clustering. Again, in presence of the eight noise attributes, standard clustering always chose $K = 1$; in contrast, penalized clustering with automatic variable selection tended to chose $K > 1$, and the two informative attributes were most often retained. However, penalized clustering did not work as well as in the previous set-up with $P = 1000$: it chose $K = 3$ most often while keeping most of the noise attributes; the reason could be that with fewer informative attributes, this was a more difficult problem than that of the previous section. Nevertheless, compared with best subset selection (Table 4), it still worked much better. In best subset selection, if we considered all possible non-null models (i.e., all possible non-null combinations of attributes), it most often selected $K = 1$; in all cases, only one noise attribute was included in the selected model. Because there was indeed only one cluster according to any noise attribute, the choice of $K = 1$ based on any noise attribute was correct; in other words, the selected model was correct, though of no interest. This highlights a unique point that in variable selection for clustering, unlike in regression, a correct model for the data based on a subset of attributes (e.g., noise attributes here) may be of no interest!

In addition, we considered only the models containing at least two attributes in best subset selection (Table 4). It turned out that all the selected models included only two attributes; it was still more likely to choose $K = 1$, most often with two noise attributes, which was again caused by the the complication of having so many correct models of no interest: there was indeed only one cluster based on any two of the noise attributes. In summary, we concluded that subset selection was not suitable at all for variable selection in clustering, whereas penalized clustering was much more effective.

3.2 Tumor Subtype Discovery Using Gene Expression Profiles

Golub et al. (1999) studied discovering two subtypes of human acute leukemia, acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL), using microarray gene expression data. Distinguishing the two subtypes is clinically important because, for example, the same chemotherapy applied to ALL patients may not be suitable for AML patients. They used Affymetrix microarrays, each containing 7129 genes. We applied model-based clustering to their data with 38 patients, among which 21 were ALL while the other 11 were AML patients; each patient was treated as an observation while the genes were treated as attributes. Because most of the 7129 genes were not believed to be informative to discriminating between ALL and AML, and in fact, many of the genes

	Standard	Penalized	
K	BIC	BIC	λ
1	76966	69691	>0
2	73802	68504	5
3	*71104*	66630	3
4	72232	65378	3
5	-	64034	2
6	-	62912	2
7	-	*61950*	2
8	-	63626	3

Table 5: BIC values for various numbers (K) of clusters in standard and penalized clustering for Golub' gene expression data.

	Standard			Penalized						
Samples/clusters	1	2	3	1	2	3	4	5	6	7
ALL	4	0	23	0	1	1	1	7	8	9
AML	0	4	7	6	0	0	0	4	1	0

Table 6: Clustering results for Golub's data.

were not even expressed in any sample, we filtered out most genes: we ranked the genes based on their sample variances across all 38 samples, and used only the top 2000 ones. For each method, we started from $g = 1$ and increased g until a minimum BIC was reached. The standard clustering chose $K = 3$ while the penalized one selected $K = 7$ (Table 5).

The clustering results are detailed in Table 6. The penalized method performed better than the standard clustering: the former incorrectly assigned five while the latter misclassified seven AML samples into the clusters with the ALL samples as the majority. In penalized clustering, although 35% of the mean parameter estimates were 0's, only eight genes had their cluster-specific mean estimates as 0's across all seven clusters, and hence were regarded as non-informative; previous studies demonstrated that there were indeed a large number of the genes differentially expressed between ALL and AML samples (Thomas et al., 2001; Pan, 2002).

3.3 Gene Function Discovery Using Gene Expression Profiles

Because many genes still have unknown functions, a biologically important subject is to computationally predict gene functions using, for example, gene expression profiles (Brown et al., 2000). The premise is that co-expressed genes are likely to share the same biological function. Due to incomplete knowledge, it is equally important to discover new gene functions; there is functional heterogeneity within most of functional categories, and there are probably many more uncharacterized gene functions. Clustering gene expression profiles has become a popular approach for both gene function prediction and discovery (Eisen et al., 1998; Wu et al., 2002; Zhou et al., 2002; Xiao and Pan, 2005). We stress that gene function discovery is more of a clustering or unsupervised learning problem, as opposed to supervised learning: first, we do not restrict the genes of the same function to be in the same cluster/class; each of the multiple clusters of the genes coming from the same functional category may suggest some novel subcategory, a refinement of the original functional category. Second, we allow the existence of some unknown and novel classes: some genes of unknown function do not have to be predicted to have any one of the known functions because they may have some unknown new functions. Here, we considered gene function discovery using gene expression data for yeast *S cerevisiae*. Specifically, we used a gene expression data set containing 300 microarray experiments with gene deletions and drug treatments (Hughes et al., 2000). Variable selection was highly relevant here: first, as shown in simulation, incorrectly using noise attributes might degrade the performance of clustering, obscuring some interesting clustering structures; second, it was also biologically important to identify which microarray experiments (i.e., attributes) were informative in clustering, linking putative functions of gene clusters to biological perturbations underlying the microarray experiments.

One difficulty in evaluating the performance of a clustering algorithm for real data is how to choose an appropriate criterion. Although our interest was in clustering for gene function discovery, for the purpose of evaluations, we treated the problem as supervised learning: each gene had its response variable as its known function; gene functions were downloaded from the MIPS database (Mewes et al., 2004). For illustration, we only considered two gene functions, *cytoplasm* and *mitochondrion*, with 100 genes in each class as training data; we used other 406 and 212 genes in the two functional classes as test data.

We first used the training data without their class labels: we clustered the 200 gene expression profiles into, say K, clusters. Then, for each cluster, based on the class labels of the training observations assigned to the cluster, we assigned a class label or class probability to the cluster. There were two ways to do so, namely, hard classification and soft classification. For hard classification, we assign each cluster a class label that was possessed by the majority of the observations in the cluster. For a test observation that was assigned to a cluster, we predicted its class label as that of the cluster. For soft classification, for each cluster, we first calculated the proportion of the training observations in each class. Suppose P_k^c was such a proportion for class c in cluster k. For a test observation, if it was assigned to cluster k with posterior probability τ_k, it was classified to class c with probability $\sum_{k=1}^{K} P_k^c \tau_k$; summing these probabilities all over, we obtained an expected number of test observations assigned to each class.

3.3.1 USING THE ORIGINAL DATA

Both standard clustering and penalized clustering selected $K = 7$ by BIC (Table 7), with their predictive performances for the test data given in Table 8. The results were close. For penalized clustering, some MPLEs of the cluster-specific mean parameters were exactly zero; their numbers ranged from 36 to 126 in the seven clusters. However, there was no single attribute for which the mean parameter MPLEs were all zero in the seven clusters, hence all the 300 attributes were used in final clustering. This example showed that our penalized clustering performed as well as the standard clustering method for data with none or few non-informative attributes.

As a comparison, we applied the nearest shrunken centroids (NSC) (Tibshirani et al., 2003), random forests (RF) (Breiman, 2001), and support vector machines (SVM) (Vapnik, 1998), and thus treating the problem as supervised learning; the NSC was specifically developed for classification with gene expression data, while the RF and SVM are two state-of-the-art machine learning tools.

	Standard	Penalized	
K	BIC	BIC	λ
1	51420	49830	> 0
2	46993	46409	5
3	44252	44170	2
4	42674	42352	5
5	41944	41382	5
6	41891	40716	2
7	*41797*	*38368*	5
8	42138	39448	5

Table 7: BIC values for various numbers (K) of clusters in standard and penalized clustering for Hughes' gene expression data.

	Hard classification				Soft classification			
	Standard		Penalized		Standard		Penalized	
Truth	Pred=1	Pred=2	1	2	1	2	1	2
1	375	31	377	29	265.5	140.5	271.9	134.1
2	111	101	107	105	83.5	128.5	81.4	130.6
Accuracy	0.770		0.780		0.638		0.651	

Table 8: Predictions of standard clustering and penalized clustering over a separate test data set for Hughes' gene expression data. BIC selected $K = 7$ for both the methods.

	NC ($P = 300$)		NSC ($P = 6$)		RF		SVM	
Truth	Pred=1	Pred=2	1	2	1	2	1	2
1	313	93	304	102	316	90	362	44
2	61	151	68	144	48	164	81	131
Accuracy	0.751		0.725		0.777		0.798	

Table 9: Predictions over a separate test data set based on the nearest shrunken centroid without shrinkage (called NC) and with shrinkage (NSC), random forests (RF) and support vector machine (SVM) for Hughes' gene expression data.

Note that it is in general unfair to compare the predictive performance of a clustering method against that of a classification or supervised learning method; our purpose here was to use the modern classifiers as benchmarks. We used the default setting of R function `randomForest()` for the RF, and used `pamr()` and `svm()` for the NSC and SVM respectively; for the latter two, a 5-fold cross-validation was used to choose tuning parameters, such as the shrinkage parameter Δ in the NSC.

For the NSC, with the selected Δ, only six attributes remained in the final model; however, using all the attributes gave a slightly higher accuracy (Table 9). It was interesting to note that the NSC failed to perform better than either clustering with hard classification. There was some similarity between these two: if we regarded each cluster as a single class, then clustering with hard classification worked in a similar manner as the NSC. Nevertheless, a difference between the two was that, the NSC assumed only one cluster for each class, whereas clustering with hard classification allowed observations of the same class to go to different clusters. Unsurprisingly, the random forests and SVM also performed well for the data.

3.3.2 Using the Data with Added Noise

It seemed that there were none or few non-informative attributes in the gene expression data for gene function prediction. To mimic other real applications, where a large number of microarray experiments were available, of which however only a fraction are informative, we added 700 noise attributes to the gene expression data. Each noise variable was generated from a standard Normal distribution independent of each other.

Table 10 summarizes the results of model fitting. By BIC, standard clustering selected $K = 2$ whereas penalized clustering chose $K = 6$. With $K = 2$, standard clustering gave quite bad results for the test data with an accuracy only at about 50% (Table 11), while it performed much better with $K = 6$ (results not shown); in contrast, penalized clustering gave much higher accuracy rates. Note that with many noise attributes, in agreement with that in the previous simulation study, standard clustering probably under-estimated the true number of the clusters of interest at $K = 2$. In addition, a distinct advantage of penalized clustering was that it could correctly identify most non-informative attributes: among the added 700 noise attributes, penalized clustering correctly identified 508 such attributes; in total, 521 attributes whose cluster-specific means were estimated to be 0 for all the clusters in penalized clustering.

By comparison, the NSC performed poorly (Table 12). By 5-fold cross-validation, the method selected a model with only one attribute. Using all attributes (or some subsets of the attributes) in the NSC did not help either. In contrast, both the RF and SVM performed well. It was not clear

	Standard	Penalized	
K	BIC	BIC	λ
1	173221	167960	> 0
2	*172209*	164909	20
3	173663	163148	15
4	175812	161830	15
5	177712	160906	15
6	181799	*159805*	10
7	185537	159917	15

Table 10: BIC values for various numbers (K) of clusters in standard and penalized clustering for Hughes' gene expression data with added noise.

	Hard classification				Soft classification			
	Standard		Penalized		Standard		Penalized	
Truth	Pred=1	Pred=2	1	2	1	2	1	2
1	263	143	307	99	203.0	203.0	265.4	140.6
2	149	63	73	139	106.0	106.0	82.2	129.8
Accuracy	0.527		0.722		0.500		0.639	

Table 11: Predictions of standard clustering and penalized clustering over a separate test data set for Hughes' gene expression data with added noise.

	NC ($P = 1000$)		NSC ($P = 1$)		LDA		RF		SVM	
Truth	Pred=1	Pred=2	1	2	1	2	1	2	1	2
1	213	193	200	206	275	131	312	94	322	84
2	105	107	111	101	65	147	50	162	57	155
Accuracy	0.518		0.487		0.683		0.767		0.772	

Table 12: Predictions for a separate test data set based on several classifiers for Hughes' gene expression data with added noise.

why the NSC did not work in this example. One possible explanation was that there were multiple centroids for each class, contrary to the assumption of the NSC that there was only a single one for each class. Hence, we applied linear discriminant analysis (LDA), which imposed an assumption similar to that of the NSC. Although there was a warning message from the R function `lda()` (due to $P > n$), the LDA performed much better than the NSC.

4. Discussion

Penalized likelihood has been widely used in model regularization, particularly for variable selection. A general theory has been laid out, see, for example, Fan and Li (2001), but mainly in the context of regression and classification. We are not aware of any other penalized likelihood approaches to multivariate model-based clustering, which we have studied in this article. In particular, it is confirmed that with the chosen L_1 penalty function, it yields a simple thresholding, enabling automatic variable selection. Our numerical examples demonstrate the usefulness of our proposal, especially for "high dimension, low sample size" settings. In particular, our numerical studies suggest the following two points. First, clustering without variable selection may fail to uncover interesting structures underlying the data. Second, best subset selection not only is computationally infeasible for clustering high-dimensional data, but also may fail in small problems. In addition to high computational demand, a key issue with best subset selection is the lack of an appropriate model selection criterion: if a conventional criterion is adopted based on the correctness of a model, because of the existence of many correct models, the criterion will not be useful; for example, any model containing one cluster based on any noise variable or their combinations is correct, but of no interest, in clustering analysis.

The basic idea proposed here is generalizable to semi-supervised learning where some, but not all, observations have class labels. An approach to semi-supervised learning is to conduct clustering analysis (i.e., class discovery) simultaneously with supervised learning (i.e., classification) with a mixture model (McLachlan and Peel, 2002). Alexandridis et al. (2004) proposed such a semi-supervised learning approach with an application to tumor classification and class discovery. A drawback of their approach was that variable selection had to be taken prior to clustering/classification: they conducted variable selection using either supervised learning or other heuristics, then used the selected variables in the subsequent clustering/classification. Pan et al. (2006) extended the penalized likelihood approach discussed here to semi-supervised learning so that variable selection is accomplished simultaneously along with model fitting (i.e., clustering/classification). In particular, their simulation results clearly demonstrated the advantage of simultaneous variable selection and model fitting over that of separating variable selection from model fitting.

We have used a common diagonal covariance matrix for all clusters. There are several practical reasons. First, in "high dimension, low sample size" settings, which are of particular interest here, an unrestricted covariance matrix is infeasible for $P > n$; some modeling is necessary. Second, in the context of linear discriminant analysis with "high dimension, low sample size" data, it has been found, both empirically and theoretically, that a diagonal covariance matrix may work better than non-diagonal ones (Tibshirani et al., 2003; Bickel and Levina, 2004); in fact, naive Bayes classifiers are well known to work well in these settings. These results suggest a possible advantage of using a diagonal covariance matrix in clustering. Finally, a careful examination reveals that other more flexible choices of the within-cluster/class covariance matrix, for example, allowing different diagonal covariance matrices for different clusters/classes, destroy the mechanism of variable selection in model-based clustering, as the use of a common diagonal covariance matrix in the NSC for the same purpose of variable selection for classification. For example, consider an attribute x_p with distribution $N(0,1)$ or $N(0,2)$ for the two clusters respectively: although its means are equal, because of its different variances in the two clusters, it is still informative to discriminating between the two clusters. It is unclear how to realize automatic variable selection for other more general covariance matrices in penalized model-based clustering. Nevertheless, we acknowledge that it may be desirable to use more flexible covariance structures, for example, a non-diagonal covariance matrix, in some applications (McLachlan et al., 2003), and more work is needed to explore how to realize variable selection with such a choice in penalized model-based clustering.

In penalized/regularized methods, an important issue is the choice of the penalization parameter. Although cross-validation and other data-resampling methods can be adopted, due to their high computational cost and possibly sub-optimal performance (Efron, 2004), we have proposed a modified BIC as a model selection criterion. Based on the new results on degrees of freedom in the context of L_1-penalized regression (Efron et al., 2004; Zou et al., 2004), we propose counting only non-zero components of the maximum penalized likelihood estimate when calculating the effective number of parameters in BIC. Although it seemed to work well in our numerical examples, theoretical justifications and further evaluations are needed.

Acknowledgments

WP was supported by NIH grant HL65462 and a UM AHC Development grant, XS by NSF grants IIS-0328802 and DMS-0604394. WP thanks Benhuai Xie, Guanghua Xiao and Peng Wei for assistance with the gene expression data. The authors thank the two reviewers and the Action Editor for many helpful and constructive comments.

References

R. Alexandridis, S. Lin, and M. Irwin. Class discovery and classification of tumor samples using mixture modeling of gene expression data. *Bioinformatics*, 20:2546-2552, 2004.

P. J. Bickel, and E. Levina. Some theory for Fisher's linear discriminant function, "naive Bayes", and some alternatives when there are many more variables than observations. *Bernoulli*, 10:989-1010, 2004.

L. Breiman. Random forests. *Machine Learning* 45:5-32, 2001.

M. P. Brown, W. N. Grundy, D. Lin, N. Cristianini, C. W. Sugnet, T. S. Furey, M. Ares, and D. Haussle. Knowledge-based analysis of microarray gene expression data using support vector machines. *Proc Natl Acad Sci USA*, 97:262-267, 2000.

W. C. Chang. On using principal components before separating a mixture of two multivariate normal distributions. *Applied Statistics*, 32:267-275, 1983.

G. Ciuperca, A. Ridolfi, and J. Idier. Penalized maximum likelihood estimator for normal mixtures. *Scandinavian Journal of Statistics*, 30:45-59, 2003.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *JRSS-B*, 39:1-38, 1977.

B. Efron. The estimation of prediction error: covariance penalties and cross-validation. *JASA*, 99:619-632, 2004.

B. Efron, T. Hastie T, I. Johnstone I, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407-499, 2004.

M. Eisen, P. Spellman, P. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *PNAS*, 95:14863-14868, 1998.

J. Fan, and R. Li. Variable selection via nonconcave penalized likelihood and its Oracle properties. *JASA*, 96:1348-1360, 2001.

C. Fraley, and A. E. Raftery. How many clusters? Which clustering methods? - Answers via model-based cluster analysis. *The Computer Journal,* 41:578-588, 1998.

C. Fraley, and A. E. Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97:611-631, 2002.

C. Fraley, and A. E. Raftery. Bayesian regularization for normal mixture estimation and model-based clustering. Technical report 486, Dept. of Statistics, University of Washington, 2005.

J. H. Friedman, and J. J. Meulman. Clustering objects on subsets of attributes (with discussion). *J. R. Stat. Soc. Ser. B*, 66:815-849, 2004.

D. Ghosh D, and A. M. Chinnaiyan. (2002). Mixture modeling of gene expression data from microarray experiments. *Bioinformatics*, 18:275-286, 2002.

T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286:531-537, 1999.

P. J. Green. On use of the EM for penalized likelihood estimation. *J. R. Stat. Soc. Ser. B*, 52:443-452, 1990.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*. Springer, 2001.

P. D. Hoff. Discussion of 'Clustering objects on subsets of attributes' by Friedman and Meulman. *J. R. Stat. Soc. Ser. B*, 66:845-846, 2004.

P. D. Hoff. Subset clustering of binary sequences, with an application to genomic abnormality data. *Biometrics*, 61:1027-1036, 2005.

P. D. Hoff. Model-based subspace clustering. *Bayesian Analysis*, 1:321-344, 2006.

T. R. Hughes, M. J. Marton, A. R. Jones, C. J. Roberts, R. Stoughton, C. D. Armour, H. A. Bennett, E. Coffey, H. Dai, Y. D. He, M. J. Kidd, A. M. King, M. R. Meyer, D. Slade, P. Y. Lum, S. B. Stepaniants, D. D. Shoemaker, D. Gachotte, K. Chakraburtty, J. Simon, M. Bard, and S. H. Friend. Functional Discovery via a Compendium of Expression Profiles. *Cell*, 102:109-126, 2000.

A. Jasra, C. C. Holmes, and D. A. Stephens. Markov chain Monte Carlo methods and the label switching problem in Bayesian mixture modeling. *Statistical Science*, 20:50-67, 2005.

S. Kim, M. G. Tadesse, and M. Vannucci. Variable selection in clustering via Dirichlet process mixture models. *Biometrika*, 93:877-893, 2006.

H. Li, and F. Hong. Cluster-Rasch models for microarray gene expression data. *Genome Biology*, 2: research0031.1-0031.13, 2001.

J. S. Liu, J. L. Zhang, M. J. Palumbo, C. E. Lawrence. Bayesian clustering with variable and transformation selection (with discussion). *Bayesian Statistics*, 7:249-275, 2003.

O. L. Mangasarian, and E. W. Wild. Feature selection in k-median clustering. *Proceedings of SIAM International Conference on Data Mining, Workshop on Clustering High Dimensional Data and its Applications*, April 24, 2004, La Buena Vista, FL, pages 23-28.

G. J. McLachlan, R. W. Bean, and D. Peel. A mixture model-based approach to the clustering of microarray expression data. *Bioinformatics*, 18:413-422, 2002.

G. J. McLachlan, and D. Peel. *Finite Mixture Model*. New York, John Wiley & Sons, Inc, 2002.

G. J. McLachlan, D. Peel, and R. W. Bean. Modeling high-dimensional data by mixtures of factor analyzers. *Computational Statistics and Data Analysis*, 41:379-388, 2003.

H. W. Mewes, C. Amid, R. Arnold, D. Frishman, U. Guldener, G. Mannhaupt, M. Munsterkotter, P. Pagel, N. Strack, V. Stumpflen, J. Warfsmann, and A. Ruepp. MIPS: analysis and annotation of proteins from whole genomes. *Nucleic Acids Res.*, 32:D41-D44, 2004.

W. Pan. A comparative review of statistical methods for discovering differentially expressed genes in replicated microarray experiments. *Bioinformatics,* 12:546-554, 2002.

W. Pan, X. Shen, A. Jiang, and R. P. Hebbel. Semi-supervised learning via penalized mixture model with application to microarray sample classification. *Bioinformatics*, 22:2388-2395, 2006.

A. E. Raftery. Discussion of "Bayesian clustering with variable and transformation selection" by Liu et al. *Bayesian Statistics*, 7:266-271, 2003.

A. E. Raftery, and N. Dean. Variable selection for model-based clustering. *Journal of the American Statistical Association*, 101:168-178, 2006.

S. Richardson, and P. J. Green. On Bayesian analysis of mixtures with an unknown number of components. *JRSS-B*, 59:731-758, 1997.

G. Schwarz. Estimating the dimensions of a model. *Annals of Statistics*, 6:461-464, 1978.

X. Shen, and J. Ye. Adaptive model selection. *Journal of the American Statistical Association*, 97:210-221, 2002.

M. G. Tadesse, N. Sha, and M. Vannucci. Bayesian variable selection in clustering high-dimensional data. *Journal of the American Statistical Association*, 100:602-617, 2005.

J. G. Thomas, J. M. Olson, S. J. Tapscott, and L. P. Zhao. An efficient and robust statistical modeling approach to discover differentially expressed genes using genomic expression profiles. *Genome Research*, 11:1227-1236, 2001.

R. Tibshirani. Regression shrinkage and selection via the Lasso. *JRSS-B*, 58:267-288, 1996.

R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu. Class prediction by nearest shrunken centroids, with application to DNA microarrays. *Statistical Science*, 18:104-117, 2003.

V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

L. F. Wu, T. R. Hughes, A. P. Davierwala, M. D. Robinson, R. Stoughton, and S. J. Altschuler. Large-scale prediction of saccharomyces cerevisiae gene function using overlapping transcriptional clusters. *Nature Genetics*, 31:255-265, 2002.

G. Xiao, and W. Pan. Gene function prediction by a combined analysis of gene expression data and protein-protein interaction data. *Journal of Bioinformatics and Computational Biology*, 3:1371-1389, 2005.

K. Y. Yeung, and W. L. Ruzzo. Principal component analysis for clustering gene expression data. *Bioinformatics*, 17:763-774, 2001.

K. Y. Yeung, C. Fraley, A. Murua, A. E. Raftery, and W. L. Ruzzo. Model-based clustering and data transformations for gene expression data. *Bioinformatics*, 17:977-987, 2001.

X. Zhou, M. C. Kao, and W. H. Wong. Transitive functional annotation by shortest-path analysis of gene expression data. *Proc Natl Acad Sci USA*, 99:12783-12788, 2002.

H. Zou, T. Hastie, and R. Tibshirani. On the "Degrees of Freedom" of the Lasso. Technical report, Dept. of Statistics, Stanford University, 2004. Available at http://stat.stanford.edu/∼hastie/pub.htm.

Journal of Machine Learning Research 8 (2007) 1165-1195 Submitted 8/06; Revised 1/07; Published 5/07

Local Discriminant Wavelet Packet Coordinates for Face Recognition

Chao-Chun Liu
Dao-Qing Dai[*] STSDDQ@MAIL.SYSU.EDU.CN
Center for Computer Vision and Department of Mathematics
Sun Yat-Sen (Zhongshan) University
Guangzhou, 510275 China

Hong Yan[†] H.YAN@CITYU.EDU.HK
Department of Electric Engineering
City University of Hong Kong
83 Tat Chee Avenue
Kowloon, Hong Kong, China

Editor: Donald Geman

Abstract

Face recognition is a challenging problem due to variations in pose, illumination, and expression. Techniques that can provide effective feature representation with enhanced discriminability are crucial. Wavelets have played an important role in image processing for its ability to capture localized spatial-frequency information of images. In this paper, we propose a novel *local discriminant coordinates* method based on wavelet packet for face recognition to compensate for these variations. Traditional wavelet-based methods for face recognition select or operate on the most discriminant subband, and neglect the scattered characteristic of discriminant features. The proposed method selects the most discriminant coordinates uniformly from all spatial frequency subbands to overcome the deficiency of traditional wavelet-based methods. To measure the discriminability of coordinates, a new dilation invariant entropy and a maximum *a posterior* logistic model are put forward. Moreover, a new *triangle square ratio* criterion is used to improve classification using the Euclidean distance and the cosine criterion. Experimental results show that the proposed method is robust for face recognition under variations in illumination, pose and expression.

Keywords: local discriminant coordinates, invariant entropy, logistic model, wavelet packet, face recognition, illumination, pose and expression variations

1. Introduction

Face recognition (Zhao et al., 2003; Jain et al., 2004) has become one of the most active research areas in pattern recognition. It plays an important role in many application areas, such as human-machine interaction, authentication and surveillance. However, the wide-range variations of human face, due to pose, illumination, and expression, result in a highly complex distribution and deteriorate the recognition rate. It seems impractical to collect sufficient prototype images covering all the possible variations. Therefore, how to construct a small-size-training face recognizer robust to environmental variations is a challenging research issue. Wavelets have been successfully used in image

[*]. Also Department of Electric Engineering, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong, China. Dao-Qing Dai is the corresponding author.
[†]. Also School of Electrical and Information Engineering, University of Sydney, NSW 2006, Australia.

processing. Their ability to capture localized spatial-frequency information of image motivates us to use them for feature extraction. In this study, we investigate a new approach by extracting the features not sensitive to environmental changes from a wavelet packet dictionary.

Generally, feature extraction, discriminant analysis and classifying criterion are the three basic elements of a face recognition system. The performance and robustness of face recognition could be enhanced by improving these elements. Feature extraction in the sense of some linear or nonlinear transform of the data with subsequent feature selection is commonly used for reducing the dimensionality of facial image so that the extracted features are as representative as possible. A lot of work on face recognition has been carried out based on similarities analysis (P. Howland and Park, 2006; Belhumeur et al., 1997; Jiang et al., 2006; Martinez and Zhu, 2005; Vaswani and Chellappa, 2006; Xiang et al., 2006; Zhao et al., 2003). A well-known feature extraction method is called FisherFace, based on linear discriminant analysis (LDA), which linearly projects the image space to a low-dimensional subspace so as to discount environmental variations (Belhumeur et al., 1997; Fukunaga, 1990). This method is a statistical linear projection method which largely relies on the representation of the training samples. On the other hand, wavelet-based methods with no special focus on the training data have been used for feature extraction (Mallat, 1989; Coifman et al., 1992). The decomposition of the data into different frequency ranges allows us to isolate the frequency components introduced by intrinsic deformations due to expression or extrinsic factors (like illumination) into certain subbands. Wavelet-based methods prune away these variable subbands, and focus on the subbands that contain the most relevant information to better represent the data. WaveletFace (Chien and Wu, 2002) only uses the low-frequency subband to present the basic figure of an image, and ignores the efficacy of high-frequency components. Our previous study (Dai and Yuen, 2006) uses a wavelet enhanced regularized discriminant analysis after dimensionality reducing with low-pass filter to solve the small sample size problem, which is also a method based on the low frequency subband. Similarly, some other studies (Feng et al., 2000; Ekenel and Sanker, 2005; Zhang et al., 2004, 2005) employ the traditional transform (e.g., ICA, PCA, Neural Networks) to enhance the discriminant power in one or several special subbands, the latter always fuse the discriminant power in these different subbands for final classification (Ekenel and Sanker, 2005; Zhang et al., 2005). Moreover, as a generalization of the wavelet transform, the wavelet packet not only offers an attractive tool for reducing the dimensionality by feature extraction, but also allows us to create localized subbands of the data in both space and frequency domains. Saito and Coifman introduced the *local discriminant basis* (LDB) algorithm based on a best-basis paradigm to search for the most discriminant subbands (basis) that illuminates the dissimilarities among classes from the wavelet packet dictionary (Coifman and Saito, 1994; Saito and Coifman, 1994, 1995). Some studies (Saito et al., 2002; Stranss et al., 2003) constructed the modified LDB later. In Kouzani et al. (1997), the best-basis algorithm of Coifman and Wicherhauser (1992) is used to search for the wavelet packet basis for face representation. In Bhagavatula and Savvides (2005), PCA is performed in wavelet packet subbands and the subbands which generalize better across illumination variations for face recognition are sought. All the methods on these studies are based on the whole discriminant subband.

It is known that a good feature extractor for a face recognition system is claimed to select as many the best discriminant features as possible, which are not sensitive to arbitrary environmental variations. Nastar and Ayach (1996) investigated the relationships between variations in facial appearance and their deformation spectrum. They found that facial expressions and small occlusions affect the intensity manifold locally. Under frequency-based representation, only high-frequency spectrum is affected. Moreover, changes in pose or scale of a face and most illumination variations affect the intensity manifold globally, in which only their low-frequency spectrum is affected. Only a change in face will affect all frequency components (Zhang et al., 2004). So there are no special subbands whose all coordinates are not sensitive to these variations. In each subband, there may be only segmental coordinates which have enough discriminant power to distinguish different person, the remainder may be sensitive to environmental changes, but the methods based on the whole subband will also extract these sensitive features. Moreover, there may be no special subbands containing all the best discriminant features, because the features not sensitive to environmental variations are always distributed in different coordinates of different subbands locally. The methods based on the segmental subbands will lose some good discriminant features. Furthermore, in different subbands, the amount and distribution of best discriminant coordinates are always different. Many less discriminant coordinates in one subband may add up to a larger discriminability than another subband whose discriminability is added up with few best discriminant coordinates and residual small discriminant coordinates (Saito et al., 2002), then the few best discriminant coordinates will be discarded by the methods which search for the best discriminate subbands, but only the few best discriminant coordinates are needed. So the best discriminant information selection should be independent of their seated subbands, and only depends on their discriminability for face recognition. However, the methods based on the whole subband neglect the distribution of features, they are deficient to select the best discriminant features sometimes.

Moreover, how to measure the discriminability of coordinate is one crucial element of the whole algorithm. We translate it into the separability of each coordinate-loading ensemble, and propose a new dilation invariant entropy which is independent of the order of magnitude (OM), instead of deficient absolute "distance" measures. Furthermore, we construct a maximum a posterior (MAP) logistic model to produce a separability measure function which presents factually the separability of each coordinate-loading ensemble, that is, discriminability of each coordinate. Based on the new dilation invariant entropy and its derived separability measure function, any two coordinates are comparable for their discriminability, either they locate in the same subband or different subbands.

To solve the "*small sample size*" (SSS) problem, we use the complete linear discriminant analysis (CLDA) idea (Yang et al., 2005) which captures both regular and irregular discriminant information and makes a more powerful discriminator. For classifying criterion, the traditional Euclidean distance cannot measure the similarity very well when there exist illumination variations on facial images, and the cosine criterion is unsatisfactory when there exist pose and expression changes. Thus, we propose a new *triangle square ratio* criterion. Experimental results show that it can overcome the deficiency of the Euclidean distance and cosine criterion very well.

In this paper, to deal with illumination, pose and expression problems, we propose a new *local discriminant coordinates* (LDC) algorithm to select uniformly the most discriminant independent coordinates in all spatial frequency subbands for face recognition, in order to overcome the limitation of the methods based on whole subband. Experimental results show that our LDC feature extraction has almost overcome the shortcomings of the methods based on subband and improves the effect of feature extraction for face recognition under different environmental variations.

The contribution of this paper consists of the following:

- Further extension of wavelets to face recognition to deal with illumination, pose and expression problems.

- Introduction of a dilation invariant entropy and a maximum *a posterior* logistic model for selection of wavelet packet coordinates.
- Use of a new similarity criterion coupled with the nearest neighbor classifier.
- Design of a face recognition system, which solves the small sample size problem and is robust to variations in illumination, pose and expression.

The paper is organized as follows. In Section 2, the wavelet packet decomposition and the local discriminant basis algorithm will be introduced. Our proposed algorithm and the whole procedure will be presented in Section 3. In Section 4, experimental results are presented, followed by discussions and conclusion in Section 5.

2. Feature Extraction by Local Discriminant Basis

In this section, we first make a review on the wavelet packet decomposition, then the local discriminant basis (LDB) algorithm and the modified LDB algorithm are introduced.

2.1 The Wavelet Packet Decomposition

Wavelets are functions that satisfy certain mathematical requirements and are used as basis functions in representing data at different scales and time-frequency locations. Wavelets (Kouzani et al., 1997; Vaidyanathan, 1993) can be generated from a two-channel filter bank method which uses repeated filtering and downsampling to decompose signals into time-frequency subbands. The two-channel filter bank has a lowpass filter which removes the high frequencies and a highpass filter which removes the low frequencies. For the wavelet transform, only the lowpass filtered subband is further iterated. As a generalization of the wavelet transform, the two-channel filter banks are iterated over the lowpass and the highpass subbands in the wavelet packet decomposition. This generates a tree structure which provides many possible wavelet packet bases, accordingly, signals are decomposed into a time-frequency dictionary.

When dealing with images, the wavelet decomposition or the wavelet packet decomposition is first applied along the rows of the images, then their results are further decomposed along the columns. This results in four decomposed subimages L_1, H_1, V_1 and D_1. These subimages represent different frequency localizations of the original image which refer to Low-Low, Low-High, High-Low and High-High respectively. Their frequency components comprise the original frequency components but now in distinct ranges. While the process being iterated, only L_1 is further decomposed in the wavelet decomposition, but all L_1, H_1, V_1 and D_1 are further decomposed in the wavelet packet decomposition. Figure 1 shows a two-dimensional examples of a facial image for the wavelet decomposition and the wavelet packet decomposition with depth 2.

2.2 The Local Discriminant Basis (LDB) Algorithm

The *local discriminant bases* algorithm (Coifman and Saito, 1994; Saito and Coifman, 1994, 1995) uses an adjustment of dictionary, or a wavelet packet decomposition tree which offers a library of orthonormal basis localized both in space and in frequency. Before proceeding further, let us set our notations. Let $X = \{x_1, x_2, \cdots, x_N\}$ be an ensemble of training samples with K classes, $X = \bigcup_{y=1}^{K} X_y$, and $X_y = \{x_1^y, x_2^y, \cdots x_{N_y}^y\}$, where N_y is the number of samples belong to class y, and $N = \sum_{y=1}^{K} N_y$.

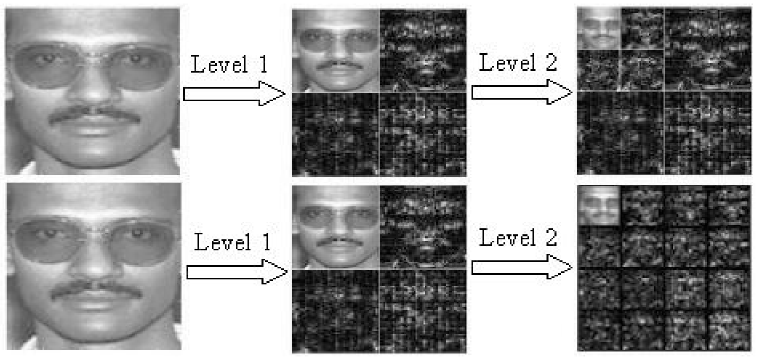


Figure 1: *(Top)* The two-dimensional wavelet decomposition of facial image with depth 2. *(Bottom)* The two-dimensional wavelet packet decomposition of facial image with depth 2.

We use $\mathcal{D}$ to represent the space-frequency dictionary consisting of a collection of wavelet packet subbands $\{B_j\}$, $j = 1, \cdots, (4^{\mathcal{L}+1} - 1)/3$, where $B_j = \{b_{j1}, b_{j2}, \cdots, b_{jn_j}\}$, $b_{ji}(i = 1, 2, \cdots, n_j)$ are wavelet packet coefficients and n_j is the size of wavelet packet subband B_j, $\mathcal{L}$ is the decomposition level of wavelet packet.

The LDB algorithm first decomposes the training samples in the dictionary $\mathcal{D}$, then sample energies at the basis coordinates are accumulated for each sample class separately to form a space-frequency energy distribution per class. Let $\Gamma^{(y)}(B_j)$ be a normalized energy of class y samples presented on the subbands B_j:

$$\Gamma^{(y)}(B_j) = (\Gamma^{(y)}(b_{j1}), \Gamma^{(y)}(b_{j2}), \cdots, \Gamma^{(y)}(b_{jn_j})) \quad \forall B_j \subset \mathcal{D}, \tag{1}$$

$$\Gamma^{(y)}(b_{jt}) \triangleq \frac{\sum_{i=1}^{N_y} |b_{jt} \cdot x_i^y|^2}{\sum_{i=1}^{N_y} \|x_i^y\|^2} \tag{2}$$

where $\cdot$ denotes the standard inner product in the Euclidean space. The loss function ϕ_1 is used to measure "distances" among K vectors $\Gamma^{(1)}(B_j), \Gamma^{(2)}(B_j), \cdots, \Gamma^{(K)}(B_j)$:

$$\phi_1(B_j) = \phi_1(\Gamma^{(1)}(B_j), \Gamma^{(2)}(B_j), \cdots, \Gamma^{(K)}(B_j)) \triangleq \sum_{\substack{m,n=1 \\ m \neq n}}^{K} d^*(\Gamma^{(m)}(B_j), \Gamma^{(n)}(B_j)) \tag{3}$$

where $d^*(\cdot, \cdot)$ is a "distance" measure, it can be the l^2 distance, the relative entropy, or the J-Divergence. Then $\phi_1(B_j)$ will be a measure of efficacy of the subband B_j for classification, and local discriminant basis are selected by the best-basis algorithm (Coifman and Wicherhauser, 1992) using the following criterion:

$$\Psi = \arg\max_{B_j \in \mathcal{D}} \phi_1(B_j). \tag{4}$$

The final step is to construct traditional discriminant analysis (e.g., LDA, CT) with features derived from the LDB feature extraction.

2.3 The Modified LDB (MLDB) Algorithm

In Saito et al. (2002), a modified version of the LDB algorithm is introduced using the empirical probability distributions instead of the space-frequency energy distribution as their selection strategy to eliminate some less discriminant coordinates in each subband locally. Let

$$\delta_{jt} \overset{\Delta}{=} \phi_2(\Gamma^{(1)}(b_{jt}), \Gamma^{(2)}(b_{jt}), \cdots \Gamma^{(K)}(b_{jt})) = \sum_{\substack{m,n=1 \\ m \neq n}}^{K} d^*(\Gamma^{(m)}(b_{jt}), \Gamma^{(n)}(b_{jt})) \tag{5}$$

that is, the discriminability of coordinate b_{jt} $(t = 1, 2, \cdots, n_j)$. Then the measure of the discriminability of B_j is obtained by summing only the $n_0(< n_j)$ largest terms, that is,

$$\phi_2(B_j) \overset{\Delta}{=} \sum_{t=1}^{n_0} \delta_{j(t)} \tag{6}$$

where $\{\delta_{j(t)}\}$ is the decreasing rearrangement of $\{\delta_{jt}\}$, and local discriminant basis are selected by the best-basis algorithm using the criterion (4) as LDB. The final step is the same as LDB.

Although the MLDB algorithm may overcome some limitations of LDB, the selection of coordinates is only limited to each subband so that coordinates in different subbands are still incomparable.

3. The General Framework of the LDC Algorithm

Our LDC algorithm uses a ternary architecture similar to LDB. We use the wavelet packet feature extraction at the first step. The main difference between LDB and our LDC algorithm is the nature of “distance” measure and feature selection strategy. We propose a new dilation invariant entropy to take the place of traditional traditional absolute “distance” measures. This ensures that the comparison of discriminability among all coordinates is independent of spatial frequency subbands. Thus, our selection can be based on all coordinates of the dictionary, but not the subbands themselves.

Moreover, LDB uses only the between-class difference, and ignores the within-class difference. This may lead to an unsatisfactory discriminability. The solution presented in this paper makes use of the maximum a posterior (MAP) logistic model. Its derived separability measure function will get a contrastive term to ensure not only the within-class difference is low, but also the between-class difference is large. Our LDC algorithm does not need the best-basis algorithm (Coifman and Wicherhauser, 1992) used in LDB, it ensures that we can select the most discriminant features without any impact of the best-basis algorithm. Subsequently, the LDC algorithm uses the complete linear discriminant analysis (CLDA) to solve the “*small sample size*” (SSS) problem, instead of the traditional LDA or CT in LDB. Finally, we modify the Euclidean distance and the cosine criterion in the nearest neighbor classifier, and replace them with the triangle square ratio criterion for classification.

3.1 The Wavelet Packet Decomposition in our LDC Algorithm

In the LDC algorithm, the wavelet packet technique is used to decompose an image into subbands that are localized in both space and frequency domains, and offers a choice of optimal coordinates for the representation of a human face. Therefore, it is possible to seek the most discriminative coordinates for classification. Because each child subband is derived from its parent subband at the above level, the coordinates in the two levels are linearly dependent. In the first experiment in

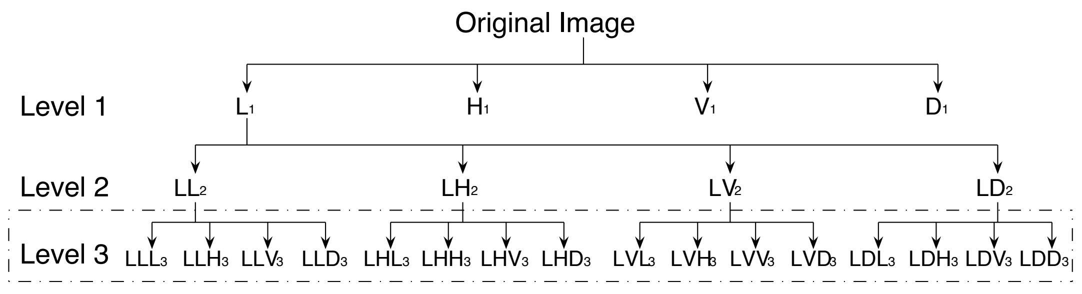


Figure 2: The wavelet decomposition tree used in this study. The *dashed part* is the spatial-frequency dictionary $\mathcal{D}$ in the LDC algorithm

Section 4, we will search for the most discriminant level by the best performance of its selected coordinates. Because it is more time-consuming when the decomposition level $\mathcal{L}$ is larger than 4, $\mathcal{L} = 4$ will be used in the experiment. In the first level, four subband images—L_1, H_1, V_1, D_1—are obtained. However, the high frequency H_1, V_1, D_1 are sensitive to noises in facial images, and Ekenel and Sanker (2005) claimed that they have low performance for classification. Moreover, the results in Table 4 show that our dilation invariant entropy used in the LDC algorithm may extract few high frequency components which may slightly affect the performance, also for the sake of computational efficiency, the H_1, V_1, D_1 components are not further decomposed. Our experimental results show that Level 3 has better performance than Level $1, 2, 4$, and the same results are also presented in Chien and Wu (2002). In fact, with the further wavelet packet decomposition, more fine scale information which may have good discriminant power is generated, however, the resolution of subband images becomes lower so that less information exists for the purpose of object localization (Grewe and Brooks, 1997). Neither little scale information nor little localization information can generate a judicious combination which has best discriminate power, so Level 3 which may give a suitable tradeoff between scale information and localization information is used in some studies (Chien and Wu, 2002; Feng et al., 2000). We also use Level 3 in the LDC algorithm, and our spatial-frequency dictionary $\mathcal{D}$ consists of 16 subbands in Level 3 (a subset of the dictionary in the LDB algorithm) (see Figure 2). The *Daubechies db*4 wavelet will be used for image decomposition (Daubechies, 1990), if the sizes of facial images are not the dyadic numbers, we will apply zero-padding extension to create the smallest dyadic images for the wavelet packet decomposition.

3.2 The Dilation Invariant Entropy

In this subsection, we first point out the deficiency of absolute "distance" measures in wavelet-based methods, then introduce our dilation invariant entropy and its property.

3.2.1 DEFICIENCY OF ABSOLUTE "DISTANCE" MEASURES IN WAVELET-BASED METHODS

To introduce our new dilation invariant entropy, we first list several traditional discriminant measures. Given two nonnegative sequences $w = (w_1, w_2, \cdots, w_n)$, $z = (z_1, z_2, \cdots, z_n)$, the *square* ℓ^2-

norm is defined by

$$d^{s\ell^2}(w,z) \triangleq \|w-z\|_2^2 = \sum_{i=1}^{n}(w_i - z_i)^2. \tag{7}$$

Suppose $\sum_{i=1}^{n} w_i = 1, \sum_{i=1}^{n} z_i = 1$, then the *Kullback-Leibler divergence* (Kullback and Leibler, 1951), also known as *relative entropy*, is defined by

$$d^{KLD}(w,z) \triangleq \sum_{i=1}^{n} w_i \log \frac{w_i}{z_i} \tag{8}$$

with the convention that $\log 0 = -\infty$, $\log \gamma/0 = \infty$ for $\gamma > 0$ and $0(\pm\infty) = 0$. A symmetric version of d^{KLD} is the *J-Divergence* (Kullback and Leibler, 1951) given by

$$d^{JDIV}(w,z) \triangleq \frac{d^{KLD}(w,z) + d^{KLD}(z,w)}{2}. \tag{9}$$

It is easy to show that measures in Equations (7)-(9) are *additive discriminant measure*, that is,

$$d^*(w,z) = \sum_{i=1}^{n} d^*(w_i, z_i) \quad {\scriptstyle (*=s\ell^2, KLD, JDIV).} \tag{10}$$

From Equations (3) and (10), we know that the discriminant measure of subband B_j in the LDB algorithm can be written as

$$\phi_1(B_j) = \sum_{\substack{m,n=1\\ m\neq n}}^{K} \sum_{t=1}^{n_j} d^*(\Gamma^{(m)}(b_{jt}), \Gamma^{(n)}(b_{jt})). \tag{11}$$

Also from Equations (5) and (6), we know that the discriminant measure of subband B_j in the MLDB algorithm can be written as

$$\phi_2(B_j) = \sum_{\substack{m,n=1\\ m\neq n}}^{K} \sum_{t=1}^{n_0} d^*(\Gamma^{(m)}(b_{j(t)}), \Gamma^{(n)}(b_{j(t)})), \; (n_0 < n_j) \tag{12}$$

where $b_{j(t)}$, $(t = 1, \cdots, n_0)$ are the first n_0 coordinates with largest discriminability in subband B_j.

However, there are no normalized conditions imposed in each subband when the decomposition level $\mathcal{L} > 0$, because for each subband $B_j (j > 1)$ (B_1 is the original image)

$$\sum_{t=1}^{n_j} \Gamma^{(y)}(b_{jt}), \sum_{t=1}^{n_0} \Gamma^{(y)}(b_{j(t)}) < 1 \;\text{ and }\; \sum_{t=1}^{n_j} \Gamma^{(y)}(b_{jt}) \not\equiv C_0, \sum_{t=1}^{n_0} \Gamma^{(y)}(b_{j(t)}) \not\equiv C_1 \quad \forall y$$

where C_0, C_1 are constants independent of y and B_j. Without the normalized conditions, the absolute "distance" measures (7)-(9) will lead to a jeopardy that $\phi_1(B_j)$ and $\phi_2(B_j)$ depend absolutely on the order of magnitude (OM) of $\Gamma^{(m)}(b_{jt}), \Gamma^{(n)}(b_{jt})$ and $\Gamma^{(m)}(b_{j(t)}), \Gamma^{(n)}(b_{j(t)})$ respectively.

Unfortunately, we find that the OM of coordinate loadings make much difference between lower spatial frequency subbands and higher spatial frequency subbands. For example, the coordinate loadings in the first spatial frequency subband B_6 (=LL$_2$ in Figure 2) of the second level may vary from 0.1 to 10, and the coordinate loadings in the second spatial frequency subband B_7 (=LH$_2$ in

B_6		B_7	
$\Gamma^{(1)}(b_{6t})$	$\Gamma^{(2)}(b_{6t})$	$\Gamma^{(1)}(b_{7t})$	$\Gamma^{(2)}(b_{7t})$
1.47e-04	3.86e-04	6.72e-08	4.90e-07
8.04e-05	3.57e-04	4.13e-07	5.04e-06
2.07e-04	3.91e-04	1.32e-06	1.10e-07
1.06e-04	3.57e-04	5.45e-07	7.02e-08
1.07e-04	3.83e-04	1.43e-07	5.32e-08
2.05e-04	3.57e-04	5.20e-09	7.14e-08

Table 1: Values of $\Gamma^{(y)}(b_{6t}), \Gamma^{(y)}(b_{7t})$ $(y = 1,2;\ t = 1,\cdots,6)$

Figure 2) of the second level may vary from 0.001 to 0.01. Table 1 lists an example of some values of $\Gamma^{(y)}(b_{6t}), \Gamma^{(y)}(b_{7t})$ computed by Equation (2) in latter experiment.

From Equations (11),(12) and (2), we can deduce a bad result that the coordinates in lower spatial frequency subbands have more discriminability because of the larger OM of theirs loadings, and the coordinates in higher spatial frequency subbands have less discriminability because of the smaller OM of theirs loadings. So the low spatial frequency subbands are dominant in the LDB and MLDB algorithm. However, it is unreasonable to neglect the middle and high spatial frequency components merely for small OM of their loadings. Our experimental results also show that not only low spatial frequency components, but also middle spatial frequency components are useful for face recognition.

3.2.2 THE DILATION INVARIANT ENTROPY AND ITS PROPERTY

First, we define that the separability of sample ensemble X is the probability of classifying all samples into their genuine classes by certain discriminant functions. It is well-known that the separability of X does not depend on the absolute distances based on the OM of sample values, but depends on the relative distances among all the samples in X. For each coordinate c, the coordinate loadings from all the training samples can induce a sample ensemble X^c in $\mathbb{R}^1$, and the discriminability of c is equivalent to the separability of X^c, so it is independent of the OM of coordinate loadings, and only depends on the relative distances among the coordinate loadings from all the training samples. Obviously, the "distance" measures used in LDB do not take this fact into account. So we propose a new "distance" measure derived from the J-Divergence. We call it the *dilation invariant entropy* :

$$d^{DIE}(w,z) \triangleq \sum_{i=1}^{n} \frac{1}{2(w_i+z_i)}\left(\frac{w_i}{z_i}\log w_i + \frac{z_i}{w_i}\log z_i\right) = \sum_{i=1}^{n} \frac{(w_i - z_i)(\log w_i - \log z_i)}{2(w_i+z_i)} \tag{13}$$

where $w = (w_1, w_2, \cdots, w_n), z = (z_1, z_2, \cdots, z_n)$ are two nonnegative sequences, with the convention that $\log 0 = -\infty$, $\log\gamma/0 = \infty$ for $\gamma > 0$ and $0(\pm\infty) = 0$.

Proposition 1 *The new relative entropy defined by Equation (13) is dilation invariant.*

Proof Suppose the dilation transform $f : w \in \mathbb{R}^n \to f(w) = aw \in \mathbb{R}^n$, $a(> 0)$ is a dilation constant, then

$$\begin{aligned}
d^{DIE}(f(w),f(z)) = d^{DIE}(aw,az) &= \sum_{i=1}^{n} \frac{(aw_i - az_i)(\log aw_i - \log az_i)}{2(aw_i + az_i)} \\
&= \sum_{i=1}^{n} \frac{a(w_i - z_i)(\log w_i + \log a - \log z_i - \log a)}{2a(w_i + z_i)} \\
&= \sum_{i=1}^{n} \frac{(w_i - z_i)(\log w_i - \log z_i)}{2(w_i + z_i)} \\
&= d^{DIE}(w,z).
\end{aligned}$$

■

In fact, the new dilation invariant entropy is the generalization of the J-Divergence (when the J-Divergence satisfies the constraint: $w+z=1$) because

$$\begin{aligned}
d^{DIE}(w,z) &= \sum_{i=1}^{n} \frac{(w_i - z_i)(\log w_i - \log z_i)}{2(w_i + z_i)} \\
&= \sum_{i=1}^{n} \frac{1}{2}\left(\frac{w_i}{w_i+z_i} - \frac{z_i}{w_i+z_i}\right)\left(\log\frac{w_i}{w_i+z_i} - \log\frac{z_i}{w_i+z_i}\right) \\
&= \frac{1}{2}\left(\sum_{i=1}^{n} w'_i \log\frac{w'_i}{z'_i} + \sum_{i=1}^{n} z'_i \log\frac{z'_i}{w'_i}\right) \\
&= d^{JDIV}(w',z')
\end{aligned} \tag{14}$$

where $w'_i = \frac{w_i}{w_i+z_i}, z'_i = \frac{z_i}{w_i+z_i}, w'_i + z'_i = 1$ and $w' = (w'_1, \cdots, w'_n), z' = (z'_1, \cdots, z'_n)$. Equation (14) shows that the dilation invariant entropy normalizes the sample ensembles into unit sample ensembles with the sum formalism, so different sample ensembles are comparable for their separability. Similarly, we can define the dilation invariant ℓ^2 norm as

$$d^{DI\ell^2}(w,z) = \left(\sum_{i=1}^{n}\left(\frac{w_i}{w_i+z_i} - \frac{z_i}{w_i+z_i}\right)^2\right)^{\frac{1}{2}} = \left(\sum_{i=1}^{n}(w'_i - z'_i)^2\right)^{\frac{1}{2}} = d^{\ell^2}(w',z'). \tag{15}$$

In Section 4, we will conduct an experiment to test the performance of the LDC algorithm using both dilation invariant entropy and dilation invariant ℓ^2 norm.

The dilation-invariance of the new relative entropy ensures that the separability of each coordinate-loadings ensemble X^c is independent of its OM. Accordingly, the discriminability of each coordinate c can be independent of its corresponding subband. It offers a benefit that any two coordinates in the dictionary $\mathcal{D}$ are comparable for their discriminability. So all the coordinates in the dictionary can be uniformly selected by a criterion.

3.3 Feature Selection Criterion

Sometimes, maximizing the between-class difference or minimizing the within-class difference alone leads to a bad result. So in the LDC algorithm, our separability measure function will contain a term to maximize the between-class difference and minimize the within-class difference simultaneously.

3.3.1 THE MAXIMUM *a posteriori* (MAP) LOGISTIC MODEL

To select the most discriminant coordinates, we make use of the Bayesian algorithm based on minimizing the error on the training set. The Bayesian algorithm adopts a probabilistic measure of similarity based on a Bayesian MAP analysis of face differences. In the traditional methods (Wang et al., 2006; Chou, 2000), the similarity measure is used to characterize what kind of image variation is typical for the same person and what is for different persons. In this paper, the MAP similarity measure is used to choose the coordinates that make the training data set with known class labels having the minimum error. In this way the selected coordinates can make the known classification of training data set the most probable:

$$\hat{c} = \arg\max_{c} \sum_{y=1}^{K} \frac{1}{\#(X_y)} \int_X P_c(X_y|x) 1(x \in X_y) dP(x)$$

where $\#(\cdot)$ is the cardinal number, and $1(\cdot)$ is the indicator function. The posterior probability $P_c(X_y|x)$ can be rewritten as

$$P_c(X_y|x) = P_c(x|X_y)P_c(X_y)/P_c(x) = P_c(x|X_y)P_c(X_y)/P(x).$$

Since $P(x)$ is not a function of the class index and thus has no effect in the MAP decision, the needed probabilistic knowledge can be represented by the class prior distribution $P_c(X_y)$ and the conditional probability $P_c(x|X_y)$ which will be modeled by logistic functions.

Definition 1 *The prior distribution $P_c(X_y)$ is defined as*

$$\begin{aligned} P_c(X_y) &= \frac{1}{T} \frac{1}{1+\exp(-d^{DIE}(\Gamma^{(y)}(c), \Gamma^{(0)}(c)))}, \\ T &= \sum_{y=1}^{K} \frac{1}{1+\exp(-d^{DIE}(\Gamma^{(y)}(c), \Gamma^{(0)}(c)))}. \end{aligned} \tag{16}$$

Definition 2 *The conditional probability $P_c(x_i^y|X_y)(x = x_i^y)$ is defined as*

$$P_c(x_i^y|X_y) = \frac{1}{1+\exp(d^{DIE}(\Gamma_i^{(y)}(c), \Gamma^{(y)}(c))} \tag{17}$$

where $\Gamma^{(y)}(c)$ is defined by Equation (2), representing the normalized spatial-frequency energy map of class y on coordinate c, and can be thought of as the center of class y. Similar to that of LDB, we set

$$\Gamma_i^{(y)}(c) \triangleq \frac{|c \cdot x_i^y|^2}{\|x_i^y\|^2}, \qquad \Gamma^{(0)}(c) \triangleq \frac{\sum_{y=1}^{K}\sum_{i=1}^{N_y} |c \cdot x_i^y|^2}{\sum_{y=1}^{K}\sum_{i=1}^{N_y} \|x_i^y\|^2}. \tag{18}$$

$\Gamma_i^{(y)}(c)$ represents the normalized spatial-frequency energy map of sample x_i^y on coordinate c, and $\Gamma^{(0)}(c)$ represents the normalized spatial-frequency energy map of all the training samples on coordinate c, which can be considered as the center of all samples.

The properties of the probability functions $P_c(X_y)$ and $P_c(x_i^y|X_y)$ can be made clear by considering the sigmoid function:

$$f(d) = \frac{1}{1+\exp(-\gamma d + \theta)}$$

with θ normally set to zero and γ set to 1 for $P_c(X_y)$ and -1 for $P_c(x_i^y|X_y)$. When $\gamma = 1$, $f(d)$ is a monotonically increasing function, a larger $d^{DIE}(\Gamma^{(y)}(c), \Gamma^{(0)}(c)))$ means that it is more probable to separate class set X_y. Contrarily, when $\gamma = -1$, $f(d)$ is a monotonically decreasing function, a smaller $d^{DIE}(\Gamma_i^{(y)}(c), \Gamma^{(y)}(c))$ means that sample x_i^y is more likely to belong to class set X_y. In fact, the idea of MAP logistic model is derived from the Fisher criterion. Moreover, the sigmoid function can effectively allay the effect of outliers which have great effect on the Fisher criterion.

3.3.2 Separability Measure and Feature Selection Criterion

For the given training data set, the empirical probability measure $P(x)$ defined on the training data set is a discrete probability measure that assigns equal mass at each sample. We define the separability measure as

$$\begin{aligned} SM(c) &= \sum_{y=1}^{K} \frac{1}{\#(X_y)} \int_X P_c(X_y|x) 1(x \in X_y) dP(x) \\ &\approx \sum_{y=1}^{K} \frac{1}{N_y} \sum_{i=1}^{N_y} P_c(x_i^y|X_y) P_c(X_y) \\ &= \sum_{y=1}^{K} P_c(X_y) \left(\frac{1}{N_y} \sum_{i=1}^{N_y} P_c(x_i^y|X_y) \right). \end{aligned} \tag{19}$$

In fact, the separability measure defined by Equation (19) is an empirical measure. If the training samples are obtained by an independent sampling from a space with a fixed probability distribution $P_0(x)$, the empirical probability distribution $P(x)$ will converge to $P_0(x)$ in distribution as $N \to \infty$. Then the empirical measure defined on the N independent training samples will converge to the expected measure as the sample size N increases. With sufficient training samples, the empirical measure is an estimate of the expected measure. The goodness of this estimate is determined by the training sample size N and the convergence rate of the empirical probability measure $P(x)$ to the limit distribution $P_0(x)$.

Furthermore, we use the following criterion for feature selection:

Criterion: Select uniformly the first N_0 coordinates from the dictionary $\mathcal{D}$ with largest separability measure defined by Equation (19).

3.4 Discriminant Analysis

LDA (Fukunaga, 1990) is a linear statistic classification method, which tries to find a linear transform so that after its application the scatter of sample vectors is minimized within each class and the scatter of mean vectors around the total mean vector is maximized simultaneously.

Let the between-class scatter operator S_b and the within-class scatter operator S_w be:

$$S_b = \frac{1}{N} \sum_{y=1}^{K} N_y (m_y - m_0)(m_y - m_0)^T, \; S_w = \frac{1}{N} \sum_{y=1}^{K} \sum_{i=1}^{N_y} (x_i^y - m_y)(x_i^y - m_y)^T$$

where m_y is the mean of the mapped training sample of class y, and m_0 is the mean across all the mapped training samples. Then the Fisher criterion function can be defined by

$$J_1(\varphi_1) = \frac{\varphi_1^T S_b \varphi_1}{\varphi_1^T S_w \varphi_1}, \quad (\varphi_1 \neq 0, \|\varphi_1\| = 1). \tag{20}$$

The solution to maximizing $J_1(\varphi_1)$ can be found by searching for a direction which maximizes the projected class means (the numerator) while minimizing the class variances in this direction (the denominator).

However, the LDA algorithm often suffers from the "*small sample size*" (SSS) problem which exists in high-dimensional pattern recognition tasks, where the number of available samples is smaller than the dimensionality of the samples. Many methods (Mika et al., 1999; Baudat and Anouar, 2000; S. Mika and Müller, 2003; Yang, 2002) discard the discriminant information contained in the null space of S_w. But a significant result is a finding that there exists crucial discriminative information in the null space of S_w (Chen et al., 2000; Zhuang and Dai, 2007; Yang and Yang, 2003; Yu and Yang, 2001). We proposed the use of regularization (Dai and Yuen, 2003), but it involves a determination of parameters. Yang et al. (2005) proposed a complete kernel Fisher discriminant analysis algorithm which makes full use of two kinds of discriminant information, regular and irregular in kernel feature space. Its advantage is that no estimation of parameter is needed. Based on their idea, we use complete linear discriminant analysis (CLDA) in the LDC algorithm to solve the SSS problem.

In Equation (20), if the within-class scatter operator S_w is invertible, $\varphi_1^T S_w \varphi_1 > 0$ always holds for every nonzero vector φ_1, and the Fisher criterion can be directly employed to extract a set of optimal discriminant vectors. If S_w is singular, there always exist vectors satisfying $\tilde{\varphi}^T S_w \tilde{\varphi} = 0$. These vectors are from the null space of S_w ($null(S_w)$) and can be very effective if they satisfy $\tilde{\varphi}^T S_b \tilde{\varphi} > 0$ at the same time (Chen et al., 2000; Zhuang and Dai, 2007; Yang and Yang, 2003; Yu and Yang, 2001). In this case, the Fisher criterion degenerates into the following between-class scatter criterion:

$$J_2(\varphi_2) = \varphi_2^T S_b \varphi_2, \quad (\|\varphi_2\| = 1). \tag{21}$$

CLDA uses the between-class scatter criterion defined in Equation (21) to derive the irregular discriminant vectors from $null(S_w)$, while using the standard Fisher criterion defined in Equation (20) to derive the regular discriminant vectors from $range(S_w)$.

In our experiments, we capture all the regular discriminant vectors which satisfy $J_1(\varphi_1) > 0$ from the range space of S_w, simultaneously, we capture all the irregular discriminant vectors which satisfy $J_2(\varphi_2) > 0$ from the null space of S_w.

3.5 A New Criterion for the Nearest Neighbor (NN) Classifier

After the discriminant features are extracted, a remaining key element of face recognition is to design a robust classifier. Because there are large numbers of classes in face recognition problems, we do not use the hyperplane classifier, but the NN classifier which is more suitable for such many-class problems. Because the NN classifier forms class boundaries with piecewise linear hyperplanes, any classifying border can be approximated by a series of hyperplanes defined locally. Moreover, classifying criterion is the core of the design. The Euclidean distance is the most popular one which exhibits the distance between two vectors intuitively. However, it ignores the correlation which is also important for measuring the similarity of two vectors. On the contrary, the cosine criterion (Zhang and Korfhage, 1999) exhibits the correlation, but ignores the distance between two vectors. In order to improve the Euclidean distance and cosine criterion, we propose a new *triangle square* *ratio* which takes into account of both distance and correlation between two vectors. Suppose υ_1, υ_2 are two vectors, the *triangle square ratio* is defined as

$$TSR(\upsilon_1, \upsilon_2) = \frac{\|\upsilon_1 - \upsilon_2\|_2^2}{\|\upsilon_1\|_2^2 + \|\upsilon_2\|_2^2}.$$

The triangle square ratio is a similarity measure based on the argument and modulus of each vector, as shown in proposition 2.

Proposition 2 *Suppose* θ *is the include angle of* υ_1 *and* υ_2*, then* $TSR(\upsilon_1, \upsilon_2) \to 0$ *if and only if* $\|\upsilon_1\|_2 \to \|\upsilon_2\|_2$ *and* $\theta \to 0$*, which implies the correlation between* υ_1 *and* υ_2 *should approach* 1.

Proof

$$\begin{aligned} TSR(\upsilon_1, \upsilon_2) &= 1 - \frac{2\|\upsilon_1\|_2 \cdot \|\upsilon_2\|_2}{\|\upsilon_1\|_2^2 + \|\upsilon_2\|_2^2} \cos\theta \quad \text{(by the cosine law)} \\ &\geq 1 - \cos\theta \quad \text{(“=” holds if and only if} \quad \|\upsilon_1\|_2 = \|\upsilon_2\|_2) \\ &\geq 0 \quad \text{(“=” holds if and only if} \quad \theta = 0). \end{aligned} \tag{22}$$

■

In fact, if υ_1 and υ_2 are unit vectors, the triangle square ratio is equivalent to Euclidean distance. Also, Equation (22) shows that triangle square ratio is a modification of cosine criterion by the term $\frac{2\|\upsilon_1\|_2 \cdot \|\upsilon_2\|_2}{\|\upsilon_1\|_2^2 + \|\upsilon_2\|_2^2}$. If $\|\upsilon_1\|_2 = \|\upsilon_2\|_2$, it is equivalent to *cosine* criterion. Moreover, we have done large numbers of numerical experiments which exclusively show that $\sqrt{TSR(\upsilon_1, \upsilon_2)}$ satisfies the triangle inequality, and the symmetric and positive definitive properties are obvious. So we guess $\sqrt{TSR(\upsilon_1, \upsilon_2)}$ is a distance measure, its proof in theory is an open problem.

Experimental results in Section 4 show that the triangular square ratio is more robust against illumination variations than the Euclidean distance, whilst retaining the robustness against pose and expression changes as the Euclidean distance. On the other hand, although the triangular square ratio marginally underperforms the cosine criterion when there are variations of illumination, it can obviously outperform the cosine criterion when there are changes of pose and expression.

3.6 The Procedure of Proposed LDC Algorithm

We summarize our local discriminant wavelet packet coordinates algorithm as follows:
Step 1: The wavelet packet transform

Expand each training sample x_i^y into the spatial-frequency dictionary $\mathcal{D}$ (see Figure 2) by the wavelet packet decomposition, then x_i^y will be represented by the loadings of coordinates in $\mathcal{D}$.
Step 2: The LDC selection transform

(2.a) For each coordinate c in the dictionary $\mathcal{D}$, use the Equations (2), (18), (16), and (17) to compute its prior distribution $P_c(X_y)$ and conditional probability $P_c(x_i^y|X_y)$, whereafter, compute its separability measure defined by Equation (19), that is, its discriminability.

(2.b) Select the first N_0 coordinates from $\mathcal{D}$ with the largest discriminability. Whereupon, each training sample x_i^y can be represented by a feature vector v_i^y which is formed by the loadings of the selected coordinates. These feature vectors form a new feature space $\mathcal{F}$.
Step 3: The CLDA transform

Use the Equations (20) and (21) to construct the subspace template by the complete linear discriminant analysis (CLDA) in $\mathcal{F}$.

Step 4: Testing a new probe sample

For a new probe sample, it will be expanded into the spatial-frequency dictionary $\mathcal{D}$ by the wavelet packet decomposition, and be extracted the loadings of the corresponding coordinates selected in Step 2 to form a new feature vector v_{new}. Then v_{new} will be projected to the subspace constructed in Step 3 and classified by the nearest neighbor classifier.

Simply, the LDC algorithm can be represented as:

$$Output = T_3 \cdot T_2 \cdot T_1 \cdot Input$$

where T_1 is the wavelet packet transform, T_2 is the LDC selection transform. and T_3 is the CLDA transform.

3.7 Computational Complexity Comparison of the LDC and LDB Algorithm

The framework of the LDC algorithm is similar to LDB. We compare their computational complexity step by step:

Step 1: The wavelet packet transform

The same procedure of the LDC and LDB algorithms cost $O(N \cdot n_r n_c \cdot \mathcal{L})$, where $n_r \times n_c$ is the size of facial images, $\mathcal{L}$ is the level of the wavelet packet decomposition.

Step 2: The LDC/LDB selection transform

(2.a) For each coordinate in $\mathcal{D}_{LDC}$, the LDC algorithm needs to compute the prior distribution (16), the conditional probability (17) and its discriminability (19), so the costs of all the coordinates in $\mathcal{D}_{LDC}$ are $O(N \cdot n_r n_c) + O(K \cdot n_r n_c) + O(N \cdot n_r n_c)$. For each subband in $\mathcal{D}_{LDB}$, the LDB algorithm needs to compute the space-frequency energy distribution (2), (1) and its measure of efficacy (3). So the costs of all the subbands in $\mathcal{D}_{LDB}$ are $O(N \cdot n_r n_c \cdot \mathcal{L}) + O(K^2 \cdot n_r n_c \cdot \mathcal{L})$.

(2.b) The LDC algorithm needs to sort all the coordinates in $\mathcal{D}_{LDC}$ by their discriminability, which costs $O(n_r n_c \cdot \log_2(n_r n_c))$. The LDB algorithm needs to select the local discriminant basis from $\mathcal{D}_{LDB}$ using the best-basis algorithm, which costs $O(\mathcal{L} \cdot 4^{\mathcal{L}})$. Then both algorithms need to represent all the training samples by new feature vectors, which cost $O(N \cdot N_0)$.

Step 3: The CLDA/LDA transform

CLDA in the LDC algorithm has the same computational complexity $O((N_0)^3)$ as LDA in the LDB algorithm in the new feature space $\mathcal{F}$.

Step 4: Testing a new probe sample

A new probe sample should be transformed by T_1, T_2, T_3 with complexity $O(n_r n_c \cdot \mathcal{L}) + O(N_0) + O(N_0 \cdot N_{ev})$ and classified by the NN classifier with complexity $O(N \cdot N_{ev})$, where N_{ev} is the number of eigenvectors extracted by CLDA or LDA.

The computational complexity of Step 2 shows that the LDC algorithm is more efficient than LDB in many real applications when K is large. Table 11 also validates the fact.

4. Experiment Results

The results presented in this section are divided into five parts. First, we construct a dictionary $\mathcal{D}$ and choose a preferable N_0 for our LDC algorithm. As aforementioned, the LDC algorithm consists of the LDC based feature extraction, the complete linear discriminant analysis (CLDA) and the nearest neighbor (NN) classifier with the triangle square ratio (TSR) criterion. In the second and third parts, we evaluate the efficacy of the LDC feature extraction and the new classifying criterion respectively. The fourth part gives the performance of the whole LDC algorithm. Some further researches of the LDC algorithm are shown in the final part.

4.1 Database

1) FERET Database: The FERET database, distributed by the National Institute of Standards and Technology, consists of 14051 eight-bit grayscale images of human heads with different expressions, poses, occlusion and illuminations (Phillips et al., 2000). Two data sets of the database are used in our experiments, one is a small data set, which contains 432 images of 72 people and each individual has six images, the other is a large data set with 255 individuals, and each person has four frontal images, the datas are extracted from four different sets, namely, Fa, Fb, Fc, and duplicate (Phillips et al., 2000). There are 1020 images in this data set. All the images are aligned by the centers of eyes and mouth, and then normalized with the resolution 92×112. Some images from both data sets of the FERET database are shown in Figure 3.

2) ORL Database: The Olivetti-Oracle Research Lab (ORL) database has 40 subjects and each subject has 10 different facial views representing various expressions, small occlusion (by glasses), different scales and orientations. So there are totally 400 facial images in the database. Each image has 92×112 pixels in gray scale. Some samples are shown in Figure 4.

3) Hybrid Database: As aforementioned, the variations of the ORL database and the FERET database are very different, which lead to unequal covariance distribution. So we blend the small FERET data set and the ORL database together, in order to test the performance of the LDC algorithm when facial images have larger illumination variations and pose, expression changes (Loog and Duin, 2004). The hybrid database has 832 images of 112 persons.

4) CMU PIE Database: The CMU Pose, Illumination, and Expression (PIE) (Sim et al., 2003) database consists of 41368 images of 68 people. Each person has images captured under 13 different poses and 43 different illumination conditions and with four different expressions. In this paper, we use a subset that focuses on illumination variations with pose and expression variations in frontal

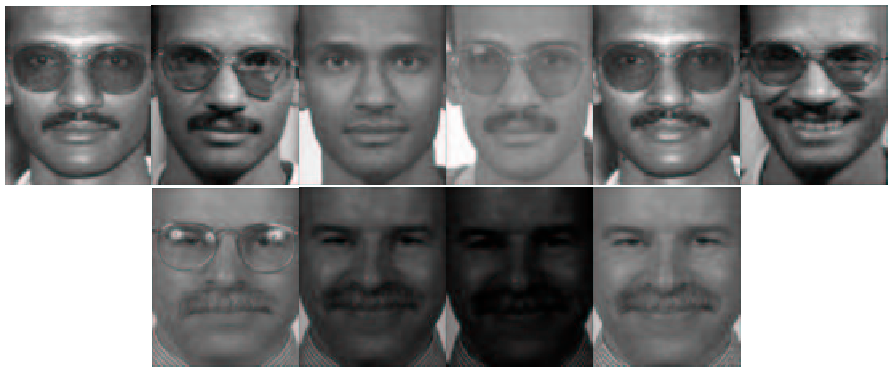


Figure 3: Facial images of the FERET database. *(Top)* A person from the small data set. *(Bottom)* A person from the large data set.

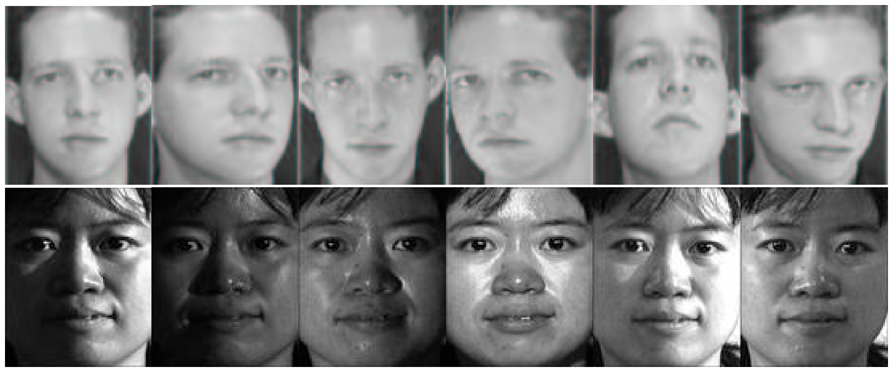


Figure 4: Segmental facial images of one person. *(Top)* From the ORL database. *(Bottom)* From the CMU PIE database.

Database	Total number of images	Number of images per person	Number of classes
ORL	400	10	40
SmallFERET	432	6	72
LargeFERET	1020	4	255
Hybrid	832	6 or 10	112
CMU-lights	2924	43	68

Table 2: Statistics for face images

view. There are 68 persons with each 43 images yielding a total of 2924 images. Each image has 92×112 pixels in gray scale. Some samples are shown in Figure 4.

The statistics of each data set is listed in Table 2.

4.2 Parameter Setting

In order to show more comparability with the PCA+CLDA, WaveletFace, LDB and MLDB algorithms and present the performance of our LDC algorithm more accurately, CLDA is used to capture the complete discriminant features in the five algorithms. The number of discriminant vectors is obtained in the same way as the LDC algorithm (see Subsection 3.4).

The FisherFace technique uses the classical PCA+LDA. The $N_{train} - K - \lambda$ (λ is the critical value which ensures S_w is non-singular) eigenvectors with largest eigenvalues are preserved on 'PCA step' (Belhumeur et al., 1997). For the PCA+CLDA algorithm, we select the first $min(N_0, M_0)$(M_0 is the number of non-zero eigenvalues) eigenvectors with largest eigenvalues on 'PCA step'. For direct LDA (DLDA), we use all the eigenvectors in their Step 2 (Yu and Yang, 2001).

The third-level lowest frequency subband LLL$_3$ with a matrix of $(n_r/8) \times (n_c/8)$ (where $n_r \times n_c$ is the resolution of original image) is referred to as WaveletFace (Chien and Wu, 2002). Because the LDB algorithm selects a best discriminant subset of the whole basis, we choose four subbands, and the number of selected coordinates is closest to N_0. For the MLDB algorithm, we choose N_0 coordinates as LDC on the scheme—five subbands with each 260 coordinates.

Methods	Parameters	
	number of features for discriminant analysis	classifier (criterion)
FisherFace	$N_{train} - K - \lambda$	NN(l^2)
PCA+CLDA	$min(N_0, M_0)$	NN(l^2)
DLDA	all eigenvectors	NN(l^2)
WaveletFace	subband LLL$_3$ in $\mathcal{D}$	NN(l^2)
LDB	four subbands	NN(l^2)
MLDB	5×260	NN(l^2)
LDC	N_0	NN(TSR)

Table 3: Parameters of aforementioned methods

In the 'Decision Step', we use the nearest neighbor classifier with the Euclidean distance for the aforementioned methods as their original forms. For our LDC algorithm, we use the new *triangle square ratio* criterion, the Euclidean distance and the cosine criterion are used for comparison. Most parameters are listed in Table 3.

The recognition rate is calculated as the ratio of the number of successful recognition and the total number of test samples. All the experiments are repeated 30 times, and the final recognition rate is the average value of the thirty results. Suppose M is the number of facial images for each person. On each database, we randomly select $i_0(< M)$ images from each person for training, while the rest $M - i_0$ images of each individual are selected for testing. i_0 is a small integer, in order to show the performance of the LDC algorithm when there are small-size-training samples.

4.3 Construction of the Dictionary $\mathcal{D}$ and Choice of N_0

In this subsection, we conduct two experiments to construct the dictionary $\mathcal{D}$ and select a suitable N_0 for the LDC algorithm.

4.3.1 CONSTRUCTION OF THE DICTIONARY $\mathcal{D}$

In the first experiment, to construct our dictionary $\mathcal{D}$, we search for the most discriminant level by the best performance of its selected coordinates. Because it is more time-consuming when the decomposition level $\mathcal{L}$ is larger than 4, $\mathcal{L} = 4$ is used in the experiment. In order to test the effect of high frequency components and show the tolerance of the dilation invariant entropy in LDC to noise, we design two schemes: Scheme 1 uses all the subbands in the wavelet decomposition tree, Scheme 2 only uses the left subtree whose root node is L$_1$, that is the H$_1$, V$_1$, D$_1$ components are not further decomposed. For each level, we select the first 1000 coordinates by the criterion in Subsection 3.3 for both schemes. Their performances on the small FERET data set and the ORL database are shown in Table 4.

Table 4 shows that the performance of Scheme 1 is marginally underperform Scheme 2, and Scheme 1 with all the subbands in the wavelet packet tree is more time-consuming than Scheme 2 which only uses the left subtree whose root node is L$_1$. So Scheme 2 is adopted in the LDC algorithm. However, the effect of high frequency components is very slight, especially in the first three levels, which implies that our dilation invariant entropy has good tolerance to noise. Table 4 also shows that the third level has the best performance, it can offer a judicious combination of

Database	Training samples (i_0)	Schemes	Level			
			1	2	3	4
ORL	3	1	82.43± 2.40	91.38 ± 2.57	92.99 ± 1.60	90.43 ± 2.38
		2	82.35± 2.54	92.19 ± 2.32	93.30 ± 1.85	92.34 ± 1.78
	4	1	85.54± 2.76	94.00 ± 1.56	94.47 ± 1.68	91.79 ± 2.08
		2	85.56± 2.87	94.43 ± 1.62	95.26 ± 1.46	94.18 ± 1.35
	5	1	87.70± 2.03	95.32 ± 1.45	95.42 ± 1.71	92.57 ± 1.62
		2	87.82± 2.12	96.03 ± 1.49	96.23 ± 1.55	94.12 ± 1.59
FERET (small)	3	1	92.42± 1.93	91.88 ± 1.69	92.33 ± 2.51	92.41 ± 2.42
		2	92.42± 1.93	91.88 ± 1.77	92.30 ± 2.43	92.53 ± 2.39
	4	1	94.93± 1.40	94.79 ± 2.25	95.23 ± 2.40	94.31 ± 2.58
		2	94.93± 1.40	94.72 ± 2.04	95.60 ± 2.20	95.30 ± 2.94
	5	1	95.93± 1.41	96.30 ± 1.76	97.04 ± 1.92	96.39 ± 2.95
		2	95.93± 1.41	96.07 ± 1.90	96.90 ± 2.33	97.13 ± 2.13

$(*)\pm(**)$: $(*)$ represents the recognition rate (%), $(**)$ represents standard deviation (%).

Table 4: Effect of high frequency components and performances of level 1,2,3,4

scale information and localization information. So Level 3 is used in the LDC algorithm, and our spatial-frequency dictionary $\mathcal{D}$ consists of the first 16 subbands in the third level (see Figure 2).

4.3.2 CHOICE OF N_0

Because all of the top N_0 coordinates are used for the classification, a natural way to determine the best N_0 is to select the top N_0 coordinates and compute the average recognition rates for various different N_0. Based on the idea, we use a global to local search strategy (Müller et al., 2001) on both data sets of the FERET database. Because the computational complexity of CLDA is $O((N_0)^3)$, for the sake of computational efficiency, we set the range [100,2500] as the original wide range of N_0. In the "global" stage, we compare the performances of the LDC algorithm using the top N_0 coordinates when N_0 increases from 100 to 2500 with interval 100 on the small data set, as shown in Figure 5 *(Left)*. It shows that the LDC algorithm has a good and stable performance after $N_0 = 700$ because more good discriminant features are used. After $N_0 = 1700$, the performance slightly decreases due to the more redundant information included. So we ascertain a more precise subrange [700, 1700] where the optimal N_0 might exist.

In the "local" stage, we compare the performances of different N_0 between 700 and 1700 with interval 100 on the large data set, as shown in Figure 5 *(Right)*. From the overall comparison of two stages, we find that when N_0 increases from 1300 to 1700 with interval 100, their performances are very close, and the performance of $N_0 = 1300$ is marginally better than others. Also for the sake of computational efficiency, we select the first $N_0 = 1300$ best discriminant coordinates in our following experiments.

However, it should point out that the choice of N_0 is not necessarily easy and needs further research. The best N_0 may be not the same for different databases, and it may be ascertained by the cross validation method. We use the natural method and generalize the same $N_0 = 1300$ to other databases, in order to show that the LDC algorithm has robustness with respect to N_0. Figure 6 also validates the conclusion.

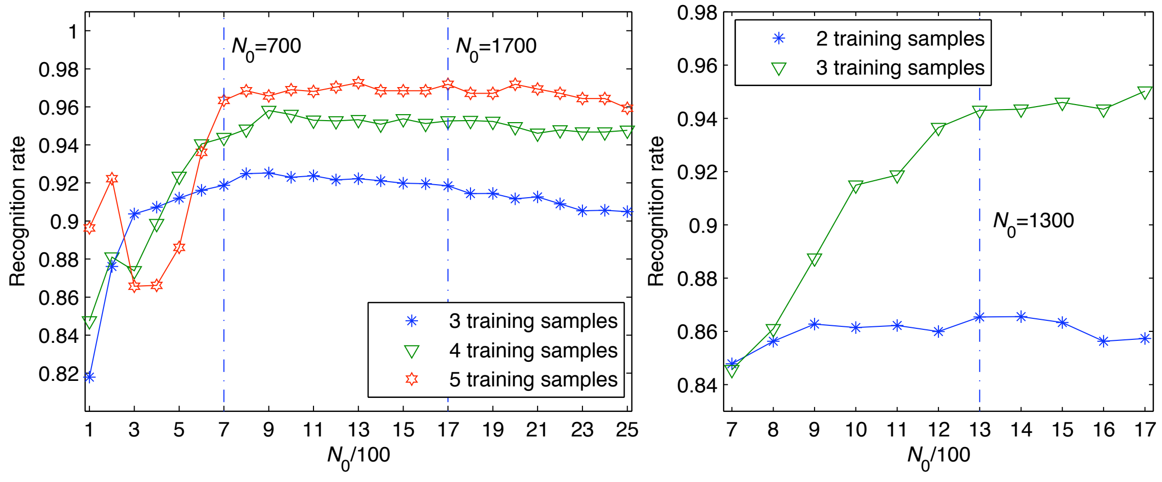


Figure 5: *(Left)* The performances of the LDC algorithm using different N_0 between 100 and 2500 with interval 100 on the small FERET data set. *(Right)* The performances of the LDC algorithm using different N_0 between 700 and 1700 with interval 100 on the large FERET data set.

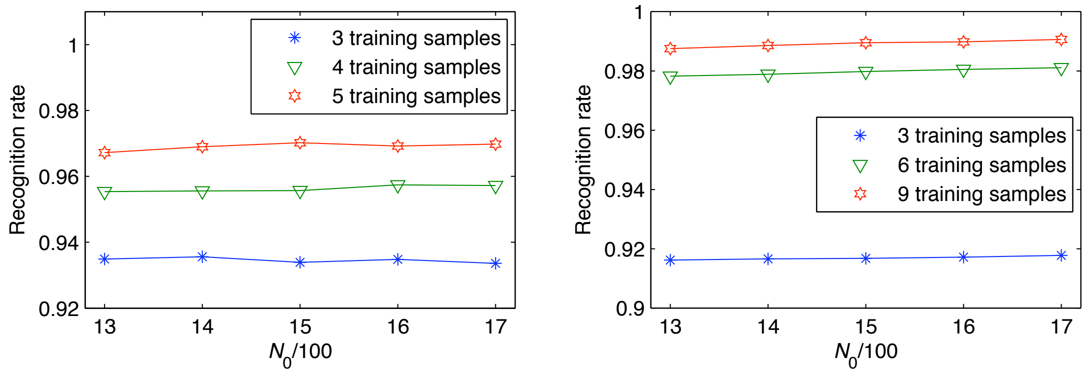


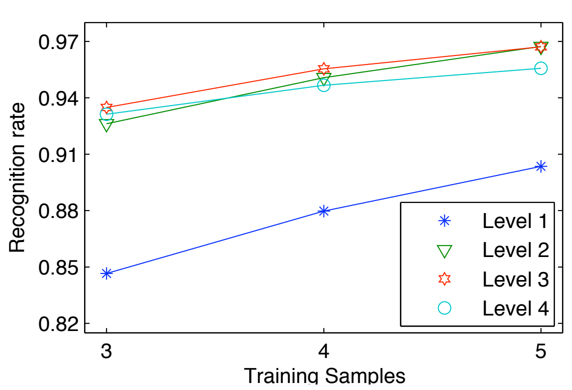
Figure 6: The performances of the LDC algorithm using different N_0 between 1300 and 1700 with interval 100. *(Left)* The ORL database. *(Right)* The CMU-lights database.

4.3.3 RECONFIRMATION OF THE DICTIONARY $\mathcal{D}$ USING N_0=1300

Moreover, we return to the anterior experiment (the construction of the dictionary $\mathcal{D}$) with the top $N_0(= 1300)$ coordinates. The results prove that the third level has the best performance once again, as shown in Figure 7.

4.4 Efficacy of the LDC Based Feature Extraction

It is of paramount importance for face recognition to extract most discriminant features that are less sensitive to environmental variations. In this subsection, we compare the efficacy of feature



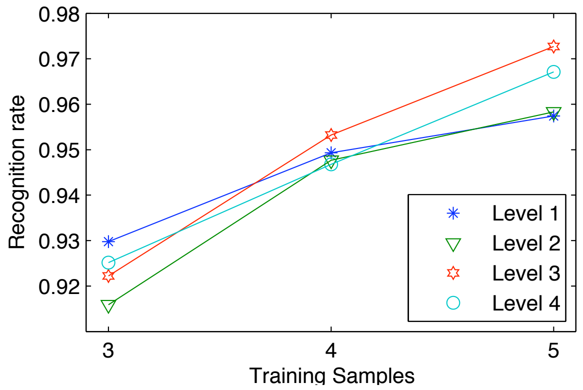


Figure 7: Performances of the LDC algorithm with N_0(=1300) coordinates in four levels. *(Left)* The ORL database. *(Right)* The small FERET data set.

Database	Training samples (i_0)	Methods with the l^2 criterion			
		LDC	LDB	MLDB	WaveletFace
ORL	3	93.43 ± 1.95	93.30 ± 2.22	92.49 ± 1.91	92.92 ± 1.60
	4	95.81 ± 1.59	95.60 ± 1.53	95.18 ± 1.39	94.56 ± 1.77
	5	96.95 ± 1.59	96.65 ± 1.20	96.43 ± 1.29	94.20 ± 1.82
FERET (small)	3	89.63 ± 3.01	87.56 ± 3.29	84.23 ± 3.26	88.80 ± 3.47
	4	92.92 ± 4.57	91.90 ± 4.48	88.80 ± 4.11	85.74 ± 4.85
	5	96.11 ± 3.39	95.69 ± 3.29	93.01 ± 3.80	87.50 ± 3.28
FERET (large)	2	79.53 ± 3.23	76.05 ± 2.90	67.16 ± 3.36	73.18 ± 4.77
	3	89.32 ± 1.82	87.45 ± 1.94	79.69 ± 1.99	72.59 ± 4.14
CMU-lights	3	79.46 ± 10.95	79.92 ± 10.91	81.30 ± 9.87	78.44 ± 10.92
	6	93.48 ± 7.63	93.69 ± 7.52	94.16 ± 6.89	85.08 ± 7.79
	9	96.83 ± 4.29	96.54 ± 4.50	96.76 ± 4.56	54.80 ± 17.52

Table 5: Comparison with wavelet-based methods

extraction in LDC with other wavelet-based methods, such as LDB, MLDB and WaveletFace on the ORL database, both data sets of the FERET database and the CMU-lights database. The setting of the feature extractions can be seen in Subsection 4.2. In order to show more comparability, *all the methods use CLDA and the NN classifier with the Euclidean distance*.

Table 5 shows that the LDC based feature extraction has the best result on the ORL database, both data sets of the FERET database, and it outperforms WaveletFace, though it underperforms LDB and MLDB marginally on the CMU-lights database. Moreover, LDC is more efficient than LDB, MLDB because of the lower computational complexity when K is large, especially on the large FERET data set ($K = 255$). On the whole, the feature extraction of LDC is more effective than the kin methods, including LDB, MLDB and WaveletFace.

Database	Training samples (i_0)	Methods with the *triangle square ratio* criterion			
		LDC	LDB	MLDB	WaveletFace
ORL	3	93.49 ± 2.19	92.93 ± 2.17	92.27 ± 1.82	92.80 ± 1.72
	4	95.54 ± 1.45	95.32 ± 1.51	94.67 ± 1.64	94.13 ± 1.82
	5	96.72 ± 1.71	96.65 ± 1.54	96.37 ± 1.41	93.60 ± 1.83
FERET (small)	3	92.22 ± 2.18	90.03 ± 2.39	87.15 ± 2.91	90.54 ± 2.29
	4	95.32 ± 1.90	94.31 ± 2.58	91.53 ± 3.31	87.78 ± 2.92
	5	97.27 ± 2.01	95.56 ± 2.56	94.17 ± 2.88	87.18 ± 2.21
FERET (large)	2	86.54 ± 2.78	84.11 ± 2.82	77.39 ± 3.34	79.01 ± 4.28
	3	94.30 ± 0.37	92.39 ± 0.48	87.87 ± 1.78	74.98 ± 5.17
CMU-lights	3	91.62 ± 6.63	91.90 ± 6.57	92.31 ± 6.37	88.67 ± 8.28
	6	97.82 ± 3.18	97.82 ± 3.25	98.14 ± 2.90	87.57 ± 6.86
	9	98.75 ± 2.09	98.47 ± 2.30	98.73 ± 2.14	55.96 ± 17.81

Table 6: Efficacy of the triangle square ratio criterion

4.5 Efficacy of the Triangle Square Ratio Criterion

Classifier and its classifying criterion are also important elements for face recognition. Generally, distance-based criterion is more robust than correlation-based criterion with respect to pose and expression changes while the contrary result is shown with respect to illumination variations. To extend the capacity covering variations of pose, expression and illumination, we have proposed the new triangle square ratio criterion in Subsection 3.5. In this experiment, *we use the same four feature extractions and CLDA as in Subsection 4.4, but the Euclidean distance is replaced by the triangle square ratio criterion for the NN classifer.* The results on the ORL database, both data sets of the FERET database and the CMU-lights database are shown in Table 6.

Comparing the results on Table 6 with Table 5 which uses the Euclidean distance, it shows that the triangle square ratio criterion performs better than the Euclidean distance considerably on both data sets of the FERET database and the CMU-lights database, while its efficacy is very close to the Euclidean distance on the ORL database. In fact, the FERET database, the CMU-lights database concern about illumination variations (light intensity and direction respectively), and the ORL database concerns about expression and pose changes. Comparison results show that the triangle square ratio criterion is more robust against illumination variations than the Euclidean distance, whilst retaining the robustness against pose and expression changes as the Euclidean distance.

Furthermore, *we replace the triangle square ratio criterion with the cosine criterion, whilst keeping the other setting*, on the ORL database and the small FERET data set. The performance of the cosine criterion is showed in Table 7. The comparison between Table 6 and Table 7 shows that the triangle square ratio criterion performs better than the cosine criterion considerably on the ORL database, although it marginally underperforms the cosine criterion on the small FERET data set.

4.6 Performance of the LDC Algorithm

In this part, we compare the performance of LDC with statistical methods, including FisherFace, PCA+CLDA, DLDA, and wavelet based methods: WaveletFace, LDB, MLDB on both individual and hybrid databases. *All the methods used for comparison keep their settings as their original forms (see Subsection 4.2).*

Database	Training samples (i_0)	Methods with the *cosine* criterion			
		LDC	LDB	MLDB	WaveletFace
ORL	3	92.54 ± 1.93	91.56 ± 2.12	90.99 ± 2.03	91.88 ± 1.73
	4	94.78 ± 1.42	94.25 ± 1.50	93.71 ± 1.71	93.46 ± 1.89
	5	96.22 ± 1.60	96.00 ± 1.49	95.53 ± 1.58	93.73 ± 1.76
FERET (small)	3	92.41 ± 1.97	90.15 ± 2.56	86.99 ± 3.00	90.82 ± 2.58
	4	95.67 ± 1.96	94.86 ± 2.31	91.85 ± 3.29	87.94 ± 3.51
	5	97.27 ± 2.01	95.56 ± 2.43	94.49 ± 2.62	87.55 ± 2.59

Table 7: Performance of the *cosine* criterion

Since our motivation is to compensate for illumination, pose and expression variation, from the properties of various databases, the ORL database is used to test moderate variations in pose and expression, the CMU-lights database to test illumination variations, the FERET database for more generic situation, and the hybrid database to test heteroscedastic class covariance distribution tolerance.

4.6.1 COMPARISON ON THE INDIVIDUAL DATABASES

The comparison of results are depicted in Table 8. Although some algorithms occasionally have better performance, LDC shows stable performance for every number of training samples per class on all databases. Especially on the large FERET data set, it outperforms FisherFace, PCA+CLDA, DLDA, WaveletFace, LDB, MLDB by 30.94%, 10.49%, 31.85%, 13.36%, 10.49%, 19.38% respectively when two samples per class are used for training, and by 21.75%, 5.95%, 31.75%, 21.71%, 6.85%, 14.61% respectively when three samples per class are used for training. In particular, LDC significantly outperforms FisherFace, DLDA, WaveletFace. From the standard derivation, we can see that LDC has better stability than other algorithms.

4.6.2 COMPARISON ON THE HYBRID DATABASE

On the hybrid database, we randomly select i_0 (i_0=2 to 4) images from each person for training, the rest ($6-i_0$) images of each individual in the small FERET data set are tested while the rest ($10-i_0$) images of each individual in the ORL database are used for testing. The comparison results are recorded in Table 9.

It shows that when the number of training sample per class increases from 2 to 4, the average recognition rates of LDC are from 82.20% to 90.28%. The performance is better than FisherFace, PCA+CLDA, DLDA, WaveletFace, LDB and MLDB which increase from 60.38%, 79.06%, 66.86%, 77.43%, 77.99%, 76.00%, to 75.63%, 90.64%, 80.85%, 66.09%, 88.11%, 86.09% respectively on the hybrid database.

As a whole, these experimental results reveal that LDC has better performance and stability on the ORL, FERET, CMU-lights databases than other methods, including FisherFace, PCA+CLDA, DLDA, WaveletFace, LDB and MLDB, especially outperforms FisherFace, DLDA, WaveletFace.

Database	Methods	Training samples (i_0)		
		3	4	5
ORL	FisherFace	87.20 ± 1.96	90.53 ± 1.87	92.07 ± 1.97
	PCA+CLDA	91.89 ± 1.97	94.81 ± 1.68	96.48 ± 1.58
	DLDA	84.17 ± 2.23	87.31 ± 1.92	90.17 ± 1.55
	WaveletFace	92.92± 1.60	94.56 ± 1.77	94.20 ± 1.82
	LDB	93.30 ± 2.22	95.60 ± 1.53	96.65 ± 1.20
	MLDB	92.49 ± 1.91	95.18 ± 1.39	96.43 ± 1.29
	LDC(TSR)	93.49 ± 2.19	95.54 ± 1.45	96.72 ± 1.71
FERET (small)	FisherFace	84.85 ± 3.64	88.01 ± 4.91	91.94 ± 4.23
	PCA+CLDA	89.07 ± 2.88	92.85 ± 4.06	95.60 ± 3.95
	DLDA	80.45 ± 4.81	86.34 ± 5.28	88.61 ± 6.41
	WaveletFace	88.80± 3.47	85.74 ± 4.85	87.50 ± 3.28
	LDB	87.56 ± 3.29	91.90 ± 4.48	95.69 ± 3.29
	MLDB	84.23 ± 3.26	88.80 ± 4.11	93.01 ± 3.80
	LDC(TSR)	92.22 ± 2.18	95.32 ± 1.90	97.27 ± 2.01
FERET (large)		Training samples (i_0)		
		2	3	
	FisherFace	55.60 ± 5.10	72.55 ± 2.57	
	PCA+CLDA	76.05 ± 3.36	88.35 ± 1.74	
	DLDA	54.69 ± 4.86	62.55 ± 2.30	
	WaveletFace	73.18± 4.77	72.59 ± 4.14	
	LDB	76.05 ± 2.90	87.45 ± 1.94	
	MLDB	67.16 ± 3.36	79.69 ± 1.99	
	LDC(TSR)	86.54 ± 2.78	94.30 ± 0.37	
CMU-lights		Training samples (i_0)		
		3	6	9
	FisherFace	80.57 ± 8.96	94.46 ± 6.44	97.36 ± 4.08
	PCA+CLDA	78.48 ± 10.89	93.91 ± 7.72	97.45 ± 3.70
	DLDA	76.92 ± 7.48	87.49 ± 6.60	92.65 ± 4.60
	WaveletFace	78.44± 10.92	85.08 ± 7.79	54.80 ± 17.52
	LDB	79.92 ± 10.91	93.69 ± 7.52	96.54 ± 4.50
	MLDB	81.30 ± 9.87	94.16 ± 6.89	96.76 ± 4.56
	LDC(TSR)	91.62 ± 6.63	97.82 ± 3.18	98.75 ± 2.09

Table 8: Comparison on the individual databases

4.7 Some Further Researches of the LDC Algorithm

In this subsection, we *keep the experimental settings in Subsection 4.2* and carry out some further researches of the LDC algorithm, including: effects of different wavelets, effects of different relative "distance" measures and comparison of CPU time.

Methods	Training samples (i_0)		
	2	3	4
FisherFace	60.38 ± 3.24	72.58 ± 2.51	75.63 ± 2.87
PCA+CLDA	79.06 ± 2.22	86.99 ± 1.34	90.64 ± 1.77
DLDA	66.86 ± 2.57	74.55 ± 2.61	80.85 ± 1.89
WaveletFace	77.43 ± 2.28	76.21 ± 1.55	66.09 ± 2.86
LDB	77.99 ± 2.21	85.81 ± 1.73	88.11± 1.78
MLDB	76.00 ± 2.07	83.41 ± 1.85	86.09 ± 2.20
LDC(TSR)	82.20 ± 1.69	88.46 ± 1.44	90.28 ± 1.47

Table 9: Comparison on the hybrid database

Database	wavelets	Training samples (i_0)		
		3	4	5
ORL	*harr*	92.49 ± 2.16	94.79 ± 1.55	96.33 ± 1.53
	db6	92.86 ± 1.52	95.24 ± 1.53	96.30 ± 1.55
	sym2	92.83 ± 2.02	95.29 ± 1.74	96.78 ± 1.77
	coif2	93.27± 1.37	94.94 ± 1.39	95.67 ± 1.67
	bior2.4	92.38 ± 2.08	94.04 ± 1.37	94.93 ± 1.54
	rbio2.4	93.71 ± 2.10	95.19 ± 1.33	95.78 ± 1.64
	db4	93.49 ± 2.19	95.54 ± 1.45	96.72 ± 1.71
FERET (small)	*harr*	93.69 ± 2.07	95.86 ± 2.17	97.22 ± 1.96
	db6	91.76 ± 2.65	94.58 ± 2.44	95.42 ± 2.42
	sym2	92.56 ± 2.41	95.72 ± 2.60	97.22 ± 2.34
	coif2	91.73 ± 2.55	94.75± 2.15	96.30 ± 1.94
	bior2.4	90.88 ± 2.10	94.47 ± 2.78	95.46 ± 1.86
	rbio2.4	91.59 ± 2.20	95.12 ± 2.37	96.16 ± 2.55
	db4	92.22 ± 2.18	95.32 ± 1.90	97.27 ± 2.01

Table 10: Performances of different wavelet basis functions

4.7.1 EFFECTS OF DIFFERENT WAVELET BASIS FUNCTIONS

We use different wavelet basis functions for the wavelet packet decomposition on the ORL database and the small FERET data set, including: *harr* wavelet, Daubechies *db6* wavelet, Symlets *sym2* wavelet, Coiflets *coif2* wavelet, Biorthogonal spline *bior2.4* wavelet, Reverse biorthogonal spline *rbio2.4* wavelet. Their performances are depicted in Table 10.

Table 10 shows that the performances of the *harr* and *sym2* wavelets are very close to the *db4* wavelet. Although other wavelets a little underperform the *db4* wavelet, their performances are also better than some other methods shown in Table 8. So we can conclude that the changes among aforementioned different wavelet basis functions have small effects on the performance of the LDC algorithm. Moreover, the orthogonal wavelets are superior to the biorthogonal wavelets in general.

4.7.2 EFFECTS OF DIFFERENT RELATIVE "DISTANCE" MEASURES

In order to show the effect of different relative "distance" measures on the performance of classification, we compare the dilation invariant entropy with the dilation invariant l^2 norm (15) on the

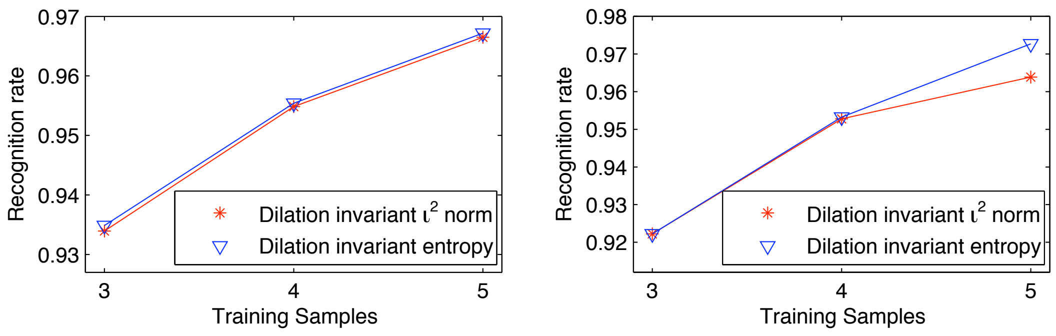


Figure 8: Performances of the LDC algorithm using the dilation invariant entropy and the dilation invariant l^2 norm. *(Left)* The ORL database. *(Right)* The small FERET data set.

Database	Methods				
	LDC	FisherFace	LDB	MLDB	WaveletFace
ORL ($K = 40$)	154	8	336	263	23
SmallFERET ($K = 72$)	164	24	415	400	40
LargeFERET ($K = 255$)	269	521	2888	3045	123

Table 11: Comparison of training CPU time (seconds)

ORL database and the small FERET data set. The results are shown in Figure 8. It shows that the dilation invariant l^2 norm marginally underperforms the dilation invariant entropy, which implies the changes between aforementioned different relative "distance" measures have slight effects on the performance of the LDC algorithm.

4.7.3 Comparison of CPU Time

We conduct an experiment to compare the time-consumption of the LDC algorithm with the popular statistics-based method: FisherFace and the wavelet-based methods: LDB, MLDB, WaveletFace on the ORL database and both data sets of the FERET database. We randomly select 3 images from each person for training. The experiments are implemented using MATLAB in a personal computer with Pentium 4 CPU and 256MB RAM. The time-consumptions are shown in Table 11. Although LDC is less efficient than WaveletFace, it is considerably more efficient than LDB and MLDB, especially when K is large. When K increases, the time-consumption of LDC increases more slowly than that of FisherFace, so that LDC can catch up with and surpass the efficiency of FisherFace.

It should point out that in our experiments, LDB selects the four best discriminant subbands from the local discriminant basis (see Subsection 4.2) and the number of selected coordinates is bigger than $N_0(= 1300)$. So LDB takes more time than LDC, MLDB in the CLDA transform. However, MLDB based feature extraction needs to estimate the probability density functions, when K increases, it takes more and more time than LDB based feature extraction, so the total time in Table 11 shows that LDB is more expensive than MLDB on the ORL database and the small FERET data set due to their small K, the contrary result is shown on the large FERET data set due to its large K.

5. Discussions and Conclusion

In this paper, we have presented a novel local discriminant coordinates (LDC) method based on wavelet packet for face recognition to compensate for illumination, pose and expression variations. The method searches for the most discriminant coordinates from a wavelet packet dictionary, instead of the most discriminant basis as in the LDB algorithm. The LDC idea makes use of the scattered characteristic of the best discriminant features. In our method, the feature selection procedure is independent of subbands, and only depends on the discriminability of all coordinates. We have shown that the traditional "distance" measures (e.g., the l^2 distance, relative entropy) are deficient to measure the separability, while comparing the separability of two sample ensembles. We have proposed a new dilation invariant entropy which is independent of the order of magnitude. We have used the dilation invariant entropy and a MAP logistic model to measure the separability of coordinate-loading ensemble accurately. It locates in either low spatial frequency subbands or high spatial frequency subbands. So any two coordinates in the wavelet packet dictionary are comparable for their discriminability. Experimental results show that the LDC based feature extraction is more effective than LDB, MLDB, WaveletFace, PCA for feature extraction.

The LDC based feature extraction not only selects low frequency components, but also middle frequency components. From its significant improvement upon the WaveletFace method which only uses low frequency components, we can conclude that middle frequency components are helpful for face recognition, since their judicious combination with low spatial frequency components can improve the performance of face recognition greatly.

We have modified the Euclidean distance and the cosine criterion in the nearest neighbor classifier, and proposed a new triangle square ratio criterion which takes into account of two similarity measures, distance and correlation. Experimental results show that the triangle square ratio criterion is more robust against illumination variations than the Euclidean distance, while retaining the robustness against pose and expression changes as the Euclidean distance. Also, it can obviously outperform the cosine criterion when there are changes of pose and expression, although it marginally underperforms the cosine criterion when there are variations of illumination. So it can well extend the capacity covering variations of pose, expression and illumination.

We have used the ORL database to test moderate variations in pose and expression, the CMU database to test illumination variations, the FERET database for more generic situation, and the hybrid to test heteroscedastic class covariance distribution.

In conclusion, experimental results show that our LDC algorithm can well preserve the most discriminant information of facial image and improve the performance for face recognition under different variations. Also, experimental results show that our LDC algorithm has robustness with respect to the number of selected coordinates. The changes among some different wavelet basis functions and different relative "distance" measures have few effects on its performance. Moreover, it is an efficient method.

The LDC idea may have numerous applications beyond the one described in this paper. It also can be applied to feature or variable selection in other dictionaries of basis functions instead of wavelets, such as the local trigonometric functions.

Acknowledgments

The authors are grateful to the anonymous reviewers for their valuable comments and suggestions, which improve the paper. This project is supported in part by NSF of China (60575004, 10231040), NSF of GuangDong(05101817), the Ministry of Education of China (NCET-04-0791) and the Hong Kong Research Grant Council (CityU 122506).

References

G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12(10):2385–2404, 2000.

P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces versus Fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, Jul. 1997.

R. Bhagavatula and M. Savvides. PCA vs. automatically pruned wavelet-packet PCA for illumination tolerant face recognition. In *Fourth IEEE Workshop on Automatic Identification Advanced Technologies*, pages 69–74, 2005.

L. F. Chen, H. Y. M. Liao, J. C. Lin, M. D. Kao, and G. J. Yu. A new LDA-based face recognition system which can solve the small sample size problem. *Pattern Recognition*, 33(10):1713–1726, 2000.

J. T. Chien and C. C. Wu. Discriminant waveletfaces and nearest feature classifiers for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12):1644–1649, 2002.

W. Chou. Discriminant-function-based minimum recognition error rate pattern-recognition approach to speech recognition. In *Proceedings of IEEE*, volume 88, Aug. 2000.

R. R. Coifman and N. Saito. Constructions of local orthonormal bases for classification and regression. *Comptes Rendus Academy Science Paris*, 319:191–196, 1994.

R. R. Coifman and M. V. Wicherhauser. Entropy-based algorithm for best basis selection. *IEEE Transactions on Information Theory*, 38(2):713–718, Mar. 1992.

R. R. Coifman, Y. Meyer, and M. V. Wickerhauser. Wavelet analysis and signal processing. In M. B. Ruskai et al., editor, *Wavelets and Their Applications*, pages 153–178. Jones and Barlett, Boston, 1992.

D. Q. Dai and P. C. Yuen. Wavelet based discriminant analysis for face recognition. *Applied Mathematics and Computation*, 175:307–318, 2006.

D. Q. Dai and P. C. Yuen. Regularized discriminant analysis and its applications to face recognition. *Pattern Recognition*, 36(3):845–847, 2003.

I. Daubechies. The wavelet transform, time-frequency localization and signal processing. *IEEE Transactions on Information Theory*, 36:961–1005, 1990.

H. K. Ekenel and B. Sanker. Multiresolution face recognition. *Image and Vision Computing*, 23: 469–477, 2005.

G. C. Feng, P. C. Yuen, and D. Q. Dai. Human face recognition using PCA on wavelet subband. *Journal of Electronic Imaging*, 9(2):226–233, Apr. 2000.

K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, Boston, second edition, 1990.

L. Grewe and R. R. Brooks. On localization of objects in the wavelet domain. In *Proceedings of the 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 412–418, 1997.

A. K. Jain, A. Ross, and S. Prabhakar. An introduction to biometric recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1):4–20, 2004.

W. Jiang, G. H. Er, Q. H. Dai, and J. W. Gu. Similarity-based online feature selection in content-based image retrieval. *IEEE Transactions on Image Processing*, 15(3):702–712, 2006.

A. Z. Kouzani, F. He, and K. Sammut. Wavelet packet face representation and recognition. In *IEEE International Conference on Systems, Man, and Cybernetics*, volume 2, pages 1614–1619, Oct. 1997.

S. Kullback and R. A. Leibler. On information and sufficieny. *Annals of Mathematical Statistics*, 22:79–86, 1951.

M. Loog and R. P. W. Duin. Linear dimensionality reduction via a heteroscedastic extension of LDA: The Chernoff criterion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):732–739, 2004.

S. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, 1989.

A. M. Martinez and M. Zhu. Where are linear feature extraction methods applicable? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1934–1944, 2005.

S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K. R. Müller. Fisher discriminant analysis with kernels. In *Proceedings of the IEEE International Workshop on Neural Networks for Signal Processing IX*, pages 41–48, Aug. 1999.

K. R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, 2001.

C. Nastar and N. Ayach. Frequency-based nonrigid motion analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:1067–1079, 1996.

J. Wang P. Howland and H. Park. Solving the small sample size problem in face recognition using generalized discriminant analysis. *Pattern Recognition*, 39:277–287, 2006.

P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss. The feret evaluation methodology for face recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22: 1090–1104, 2000.

J. Weston B. Schölkopf A. Smola S. Mika, G. Rätsch and K. R. Müller. Constructing descriptive and discriminative nonlinear features: Rayleigh coefficients in kernel feature spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):623–628, 2003.

N. Saito and R. R. Coifman. Local discriminant bases. In *Proceedings of SPIE*, volume 2303, pages 2–14, 1994.

N. Saito and R. R. Coifman. Local discriminant bases and their applications. *Journal of Mathematical Imaging and Vision*, 5(4):337–358, 1995.

N. Saito, R. R. Coifman, F. B. Geshwind, and F. Warner. Discriminant feature extraction using empirical probability density estimation and a local basis library. *Pattern Recognition*, 35:2841–2852, 2002.

T. Sim, S. Baker, and M. Bsat. The cmu pose, illumination, and expression database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1615–1618, Dec. 2003.

D. J. Stranss, G. Steidl, and W. Delb. Feature extraction by shape-adapted local discriminant bases. *Signal Processing*, 83:359–376, 2003.

P. P. Vaidyanathan. Multirate systems and filter banks. *Prentice-Hall, Englewood CliCs, NJ*, 1993.

N. Vaswani and R. Chellappa. Principal components null space analysis for image and video classification. *IEEE Transactions on Image Processing*, 15(7):1816–1830, 2006.

L. W. Wang, X. Wang, and J. F. Feng. Subspace distance analysis with application to adaptive bayesian algorithm for face recognition. *Pattern Recognition*, 39:456–464, 2006.

C. Xiang, X. A. Fan, and T. H. Lee. Face recognition using recursive Fisher linear discriminant. *IEEE Transactions on Image Processing*, 15(8):2097–2105, 2006.

J. Yang and J. Y. Yang. Why can LDA be performed in PCA transformed space? *Pattern Recognition*, 36(2):563–566, 2003.

J. Yang, A. F. Frangi, J. Y. Yang, and D. Zhang. KPCA plus LDA: A complete kernel Fisher discriminant framework for feature extraction and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(2):230–243, Feb. 2005.

M. H. Yang. Kernel Eigenfaces vs. kernel Fisherfaces: Face recognition using kernel methods. In *Proceedings of Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, pages 215–220, May 2002.

H. Yu and J. Yang. A direct LDA algorithm for high-dimensional data–with application to face recognition. *Pattern Recognition*, 34(10):2067–2070, 2001.

B. L. Zhang, H. H. Zhang, and S. S. Ge. Face recognition by applying wavelet subband representation and kernel associative memory. *IEEE Transactions on Neural Networks*, 15(1):166–177, 2004.

J. Zhang and R. R. Korfhage. A distance and angle similarity measure method. *Journal of the American Society for Information Science*, 50(9):772–778, 1999.

Z. B. Zhang, S. L. Ma, and D. Y. Wu. The application of neural network and wavelet in human face illumination compensation. *Lecture Notes in Computer Science*, ISSU 3497, 2005.

W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Computing Surveys*, 35(4):399–459, 2003.

X. S. Zhuang and D. Q. Dai. Improved discriminate analysis for high dimensional data and its application to face recognition. *Pattern Recognition*, 40(5):1570–1578, 2007.

Journal of Machine Learning Research 8 (2007) 1197-1215 Submitted 8/05; Revised 2/07; Published 5/07

Synergistic Face Detection and Pose Estimation with Energy-Based Models

Margarita Osadchy RITA@CS.HAIFA.AC.IL
Department of Computer Science
University of Haifa
Mount Carmel, Haifa 31905,Israel

Yann Le Cun YANN@CS.NYU.EDU
The Courant Institute
New York University
New York, NY 10003, USA

Matthew L. Miller MLM@NEC-LABS.COM
NEC Labs America
Princeton NJ 08540, USA

Editor: Pietro Perona

Abstract

We describe a novel method for simultaneously detecting faces and estimating their pose in real time. The method employs a convolutional network to map images of faces to points on a low-dimensional manifold parametrized by pose, and images of non-faces to points far away from that manifold. Given an image, detecting a face and estimating its pose is viewed as minimizing an energy function with respect to the face/non-face binary variable and the continuous pose parameters. The system is trained to minimize a loss function that drives correct combinations of labels and pose to be associated with lower energy values than incorrect ones.

The system is designed to handle very large range of poses without retraining. The performance of the system was tested on three standard data sets—for frontal views, rotated faces, and profiles—is comparable to previous systems that are designed to handle a single one of these data sets.

We show that a system trained simuiltaneously for detection and pose estimation is more accurate on *both tasks* than similar systems trained for each task separately.[1]

Keywords: face detection, pose estimation, convolutional networks, energy based models, object recognition

1. Introduction

The detection of human faces in natural images and videos is a key component in a wide variety of applications of human-computer interaction, search and indexing, security, and surveillance. Many real-world applications would profit from *view-independent* detectors that can detect faces under a wide range of poses: looking left or right (yaw axis), up or down (pitch axis), or tilting left or right (roll axis).

In this paper we describe a novel method that can not only detect faces independently of their poses, but also simultaneously estimate those poses. The system is highly reliable, runs in real time

1. A more preliminary version of this work appears as: Osadchy et al. (2005).

on standard hardware, and is robust to variations in yaw ($\pm 90^\circ$), roll ($\pm 45^\circ$), pitch ($\pm 60^\circ$), as well as partial occlusions.

The method is motivated by the idea that multi-view face detection and pose estimation are so closely related that they should not be performed separately. The tasks are related in the sense that they could use similar features and internal representations, and must be robust against the same sorts of variation: skin color, glasses, facial hair, lighting, scale, expressions, etc. We suspect that, when trained together, each task can serve as an inductive bias for the other, yielding better generalization or requiring fewer training examples (Caruana, 1997).

To exploit the synergy between these two tasks, we train a learning machine to map input images to points in a low-dimensional space. In the low-dimensional output space we embed a *face manifold* which is parameterized by facial pose parameters (e.g., pitch, yaw, and roll). A convolutional network is trained to map face images to points on the face manifold that correspond to the pose of the faces and non-face images to points far away from that manifold. After training, a detection is performed by measuring whether the distance of the output point from the manifold is lower than a threshold. If the point is close to the manifold, indicating that a face is present in the image, its pose parameters can be inferred from the position of the projection of the point onto the manifold.

To map input images to points in the low-dimensional space, we employ a convolutional network architecture (LeCun et al., 1998). Convolutional networks are specifically designed to learn invariant representation of images. They can easily learn the type of shift-invariant local features that are relevant to face detection and pose estimation. More importantly, they can be replicated over large images (applied to every sub-windows in a large image) at a small fraction of the cost of applying more traditional classifiers to every sub-windows in an image. This is a *considerable advantage for building real-time systems*.

As a learning machine we use the recently proposed *Energy-Based Models* (EBM) that provide a description and the inference process and the learning process in a single, well-principled framework (LeCun and Huang, 2005; LeCun et al., 2006).

Given an input (an image), an Energy-Based Model associates an energy to each configuration of the variables to be modeled (the face/non-face label and the pose parameters in our case). Making an inference with an EBM consists in searching for a configuration of the variables to be predicted that minimizes the energy, or comparing the energies of a small number of configurations of those variables. EBMs have a number of advantages over probabilistic models: (1) There is no need to compute partition functions (normalization constants) that may be intractable; (2) because there is no requirement for normalization, the repertoire of possible model architectures that can be used is considerably richer. In our application we define an Energy-Based Model as a scalar-valued energy function of three variables: image, label, and pose, and we treat pose as a deterministic latent variable. Thus both label of an image and pose are inferred through the energy-minimization process.

Training an EBM consists in finding values of the trainable parameters (which parameterize the energy function) that associate low energies to "desired" configurations of variables, and high energies to "undesired" configurations. With probabilistic models, making the probability of some values large automatically makes the probabilities of other values small because of the normalization. With EBM's making the energy of desired configurations low may not necessarily make the energies of other configurations high. Therefore, one must be very careful when designing loss functions for EBMs. In our application to face detection we derive a new type of *contrastive loss function* that is tailored to such detection tasks.

The paper is organized as follows. First, some of the relevant prior works on multi-view face detection are briefly discussed. Section 2 discusses the synergy between pose estimation and face detection, and describes the basic methods for integrating them. Section 3 discusses the learning machine, and Section 4 gives the results of experiments conducted with our system. Section 5 draws some conclusions.

1.1 Previous Work

Learning-based approaches to face detection abound, including real-time methods (Viola and Jones, 2001), and approaches based on convolutional networks (Vaillant et al., 1994; Garcia and Delakis, 2002). Most multi-view systems take a *view-based* approach, which involves building separate detectors for different views and either applying them in parallel (Pentland et al., 1994; Sung and Poggio, 1998; Schneiderman and Kanade, 2000; Li et al., 2002) or using a pose estimator to select the most appropriate detector (Jones and Viola, 2003; Huang et al., 2004). Another approach is to estimate and correct in-plane rotations before applying a single pose-specific detector (Rowley et al., 1998b). Some attempts have been done in integrating pose search and detection, but in much smaller space of pose parameters (Fleuret and Geman, 2001).

Closer to our approach is that of Li et al. (2000), in which a number of Support Vector Regressors are trained to approximate smooth functions, each of which has a maximum for a face at a particular pose. Another machine is trained to convert the resulting values to estimates of poses, and a third machine is trained to convert the values into a face/non-face score. The resulting system is rather slow. See Yang et al. (2002) for survey of face detection methods.

2. Integrating Face Detection and Pose Estimation

To exploit the posited synergy between face detection and pose estimation, we must design a system that integrates the solutions to the two problems. Merely cascading two systems where the answer to one problem is used to assist in solving the other will not optimally take advantage of the synergy. Therefore, both answers must be derived from one underlying analysis of the input, and both tasks must be trained together.

Our approach is to build a trainable system that can map raw images X to points in a low-dimensional space (Figure 1). In that space, we pre-define a *face manifold* $F(Z)$ that we parameterize by the pose Z. We train the system to map face images with known poses to the corresponding points on the manifold. We also train it to map images of non-faces to points far away from the manifold. During recognition, the system maps the input image X to a point in the low dimensional space $G(X)$. The proximity of $G(X)$ to the manifold then tells us whether or not an image X is a face. By finding the pose parameters Z that correspond to the point on the manifold that is closest to the point $G(X)$ (projection), we obtain an estimate of the pose (Figure 2).

2.1 Parameterizing the Face Manifold

We will now describe the details of the parameterizations of the face manifold. Three criteria directed the design of the face manifold: (1) preserving the topology and geometry of the problem; (2) providing enough space for mapping the background images far from the manifold (since the proximity to the manifold indicates whether the input image contains a face); and (3) minimizing

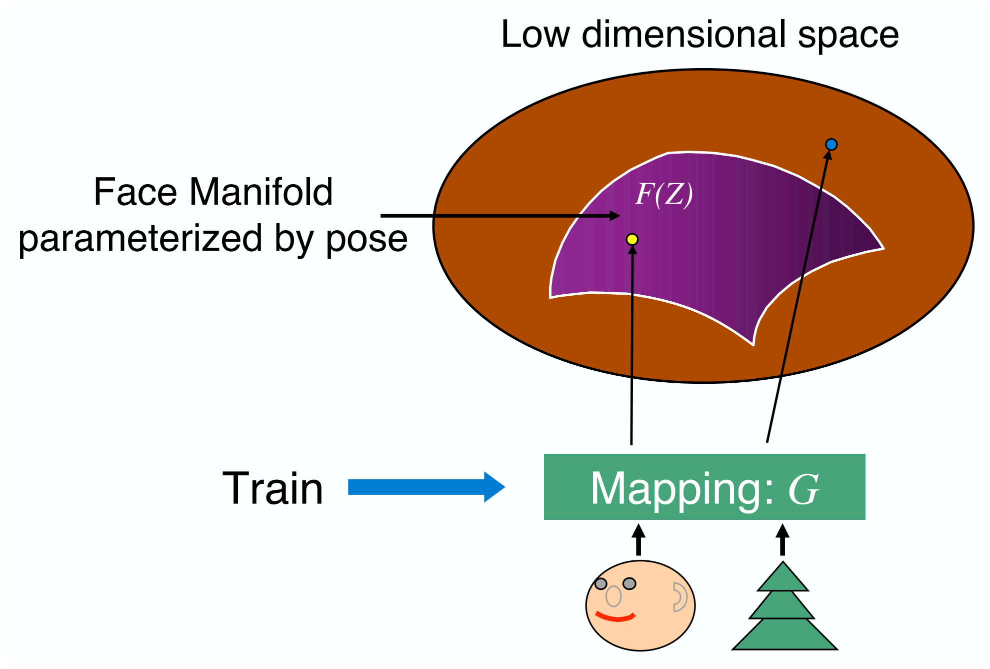


Figure 1: Manifold Mapping—Training

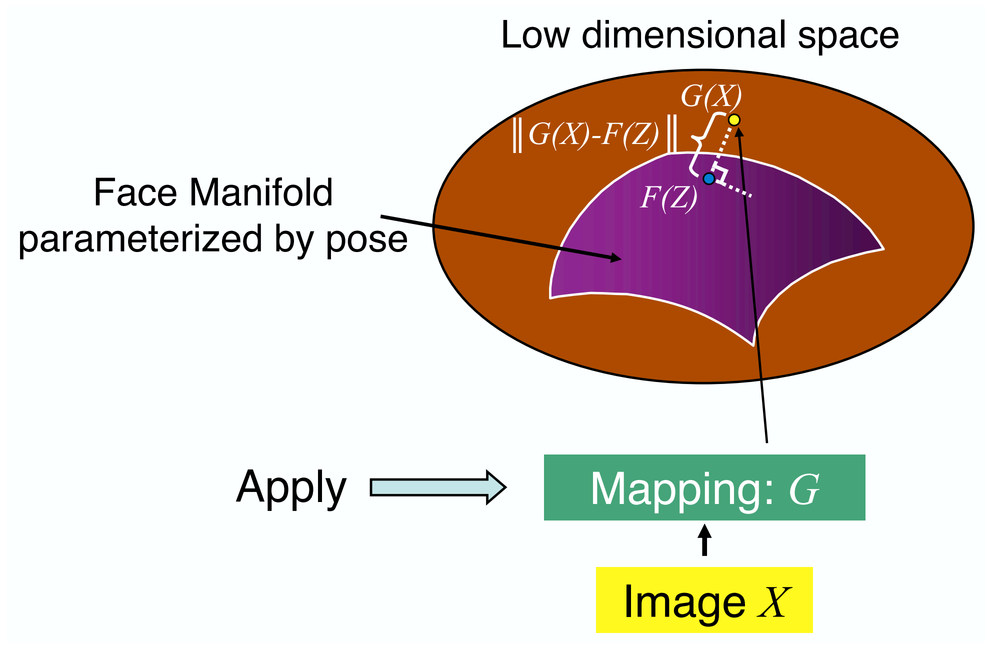


Figure 2: Manifold Mapping— Recognition and Pose Estimation.

the computational cost of finding the parameters of the closest point on the manifold to any point in the space.

Let's start with the simplest case of one pose parameter $Z = \theta$, representing, say, yaw. If we want to preserve the natural topology and geometry of the problem (the first criterion), the face manifold under yaw variations in the interval $[-90°, 90°]$ should be a half circle (with constant curvature). The natural way of representing a circle is with sine and cosine functions. In this case we embed the angle parameter in two dimensional space. Now images of faces will be mapped to points on the half circle manifold corresponding to θ, and non-face images will be mapped to points in the rest of the two dimensional space. Having lots of free space to represent non-face images may be necessary, due to the considerable amount of variability in non-face images. So increasing the dimension of embedding might help us in better separation of face and non-face images (the second criterion). In the case of single pose parameter we suggest 3D embedding.

To make the projection and parameter estimation simple (the third criterion), we embed this half-circle in a three-dimensional space using three equally-spaced shifted cosine functions (Figure 3):

$$F_i(\theta) = \cos(\theta - \alpha_i); \quad i = 1,2,3; \quad \theta = [-\frac{\pi}{2}, \frac{\pi}{2}]; \quad \alpha = \{-\frac{\pi}{3}, 0, \frac{\pi}{3}\}.$$

A point on the face manifold parameterized by the yaw angle θ is $F(\theta) = [F_1(\theta), F_2(\theta), F_3(\theta)]$. When we run the network on an image X, it outputs a vector $G(X)$. The yaw angle $\overline{\theta}$ corresponding to the point on the manifold that is closest to $G(X)$ can be expressed analytically as:

$$\overline{\theta} = \arctan \frac{\sum_{i=1}^{3} G_i(X)\cos(\alpha_i)}{\sum_{i=1}^{3} G_i(X)\sin(\alpha_i)}.$$

The point on the manifold closest to $G(X)$ is just $F(\overline{\theta})$.

The function choice is not limited to cosine. However cosines are preferable since they allow computing the pose analytically from the output of the network. Without this property, finding the pose could be an expensive optimization process, or even require the use of a second learning machine.

The same idea can be generalized to any number of pose parameters. Let us consider the set of all faces with yaw in $[-90, 90]$ and roll in $[-45, 45]$. In an abstract way, this set is isomorphic to a portion of a sphere. Consequently, we can represent a point on the face manifold as a function of the two pose parameters by 9 basis functions that are the cross-products of three shifted cosines for one of the angles, and three shifted cosines for the other angle:

$$F_{ij}(\theta, \phi) = \cos(\theta - \alpha_i)\cos(\phi - \beta_j); \quad i, j = 1,2,3.$$

For convenience, we rescale the roll angles to the range $[-90, 90]$ which allows us to set $\beta_i = \alpha_i$. With this parameterization, the manifold has constant curvature, which ensures that the effect of errors will be the same regardless of pose. Given a 9-dimensional output vector from the convolutional network $G_{ij}(X)$, we compute the corresponding yaw and roll angles $\overline{\theta}, \overline{\phi}$ as follows:

$$\begin{aligned}
cc &= \textstyle\sum_{ij} G_{ij}(X)\cos(\alpha_i)\cos(\beta_j); \quad & cs &= \textstyle\sum_{ij} G_{ij}(X)\cos(\alpha_i)\sin(\beta_j);\\
sc &= \textstyle\sum_{ij} G_{ij}(X)\sin(\alpha_i)\cos(\beta_j); \quad & ss &= \textstyle\sum_{ij} G_{ij}(X)\sin(\alpha_i)\sin(\beta_j);
\end{aligned}$$
$$\overline{\theta} = 0.5(atan2(cs + sc, cc - ss) + atan2(sc - cs, cc + ss)) \quad ;$$
$$\overline{\phi} = 0.5(atan2(cs + sc, cc - ss) - atan2(sc - cs, cc + ss)) \quad .$$

The process can easily be extended to include pitch in addition to yaw and roll, as well as other parameters if necessary.

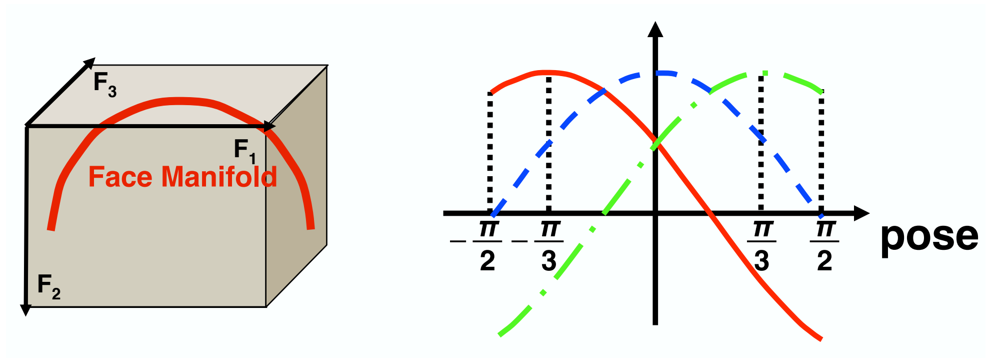


Figure 3: Left: face manifold embedding; right: manifold parametrization by single pose parameter. The value of each cosine function for one pose angle constitute the three components of a point on the face manifold corresponding to that pose.

3. Learning Machine

To map input images to points in the low-dimensional space, we employ a convolutional network architecture trained using Energy Minimization Framework. Next we present the details of the learning machine.

3.1 Energy Minimization Framework

We propose the following configuration of the Energy Based Model (LeCun and Huang, 2005; LeCun et al., 2006). Consider a scalar-valued function $E_W(Y,Z,X)$, where X is a raw image, Z is a facial pose (e.g., yaw and roll as defined above), Y is a binary label: $Y = 1$ for face, $Y = 0$ for non-face. W is a parameter vector subject to learning. $E_W(Y,Z,X)$ can be interpreted as an *energy function* that measures the degree of compatibility between the values of X,Z,Y. The inference process consists in clamping X to the observed value (the image), and searching for configurations of Z and Y that minimize the energy $E_W(Y,Z,X)$:

$$(\overline{Y},\overline{Z}) = \text{argmin}_{Y\in\{Y\},\, Z\in\{Z\}} E_W(Y,Z,X)$$

where $\{Y\} = \{0,1\}$ and $\{Z\} = [-90,90]\times[-45,45]$ for yaw and roll variables.

Ideally, if the input X is the image of a face with pose Z, then a properly trained system should give a lower energy to the face label $Y = 1$ than to the non-face label $Y = 0$ for any pose: $E_W(1,Z,X) < E_W(0,Z',X)$, $\forall Z'$. For accurate pose estimation, the system should give a lower energy to the correct pose than to any other pose: $E_W(1,Z',X) > E_W(1,Z,X)$, $\forall Z' \neq Z$. Training a machine to satisfy those two conditions for any image will guarantee that the energy-minimizing inference process will produce the correct answer.

Transforming energies to probabilities can easily be done via Gibbs distribution:

$$P(Y,Z|X) = \exp(-\beta E_W(Y,Z,X)) / \int_{y\in\{Y\},z\in\{Z\}} \exp(-\beta E_W(y,z,X))$$

where β is an arbitrary positive constant, and $\{Y\}$ and $\{Z\}$ are the sets of possible values of y and z. With this formulation, we can easily interpret the energy minimization with respect to Y and Z as

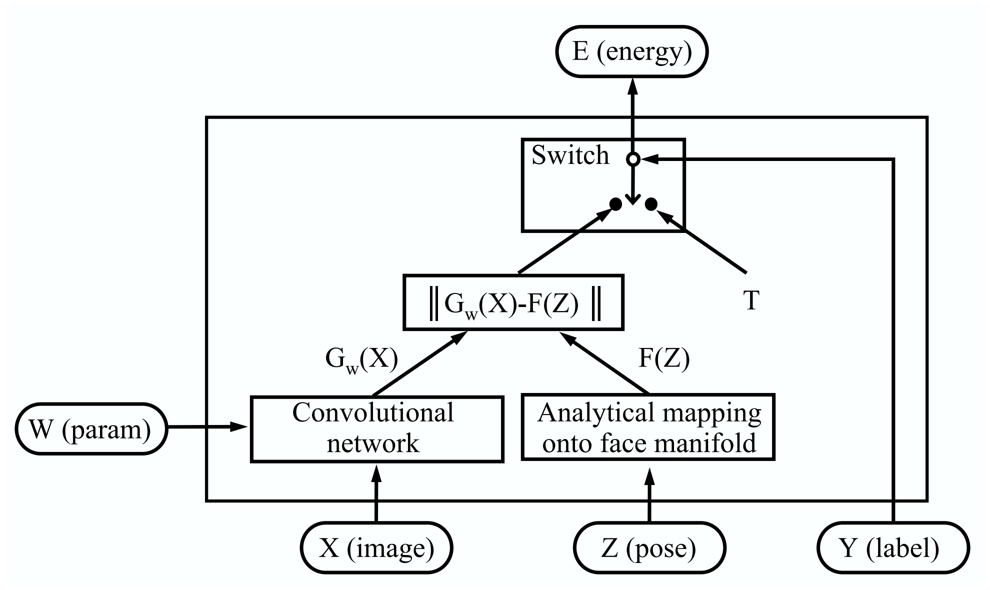


Figure 4: Architecture of the Minimum Energy Machine.

a maximum conditional likelihood estimation of Y and Z. This probabilistic interpretation assumes that the integral in the denominator (the partition function) converges. It is easy to design such an energy function for our case. However, a proper probabilistic formulation would require us to use the negative log-likelihood of the training samples as our loss function for training. This will require us to compute the derivative of the denominator with respect to the trainable parameters W. This is unnecessary complication which can be alleviated by adopting the energy-based formulation. Removing the necessity for normalization gives us complete freedom in the choice of the internal architecture and parameterization of $E_W(Y,Z,X)$, as well as considerable flexibility in the choice of the loss function for training.

Our energy function for a face $E_W(1,Z,X)$ is defined as the distance between the point produced by the network $G_W(X)$ and the point with pose Z on the manifold $F(Z)$:

$$E_W(1,Z,X) = \|G_W(X) - F(Z)\|.$$

The energy function for a non-face $E_W(0,Z,X)$ is equal to a constant T that we can interpret as a threshold (it is independent of Z and X). The complete energy function is:

$$E_W(Y,Z,X) = Y\|G_W(X) - F(Z)\| + (1-Y)T.$$

The architecture of the machine is depicted in Figure 4. Operating this machine (finding the output label and pose with the smallest energy) comes down to first finding: $\overline{Z} = \text{argmin}_{Z\in\{Z\}}||G_W(X) - F(Z)||$, and then comparing this minimum distance, $\|G_W(X) - F(\overline{Z})\|$, to the threshold T. If it's smaller than T, then X is classified as a face, otherwise X is classified as a non-face. This decision is implemented in the architecture as a *switch*, that depends upon the binary variable Y.

For simplicity we fix T to be a constant. Although it is also possible to make T a function of pose Z.

3.2 Convolutional Network

We employ a Convolutional Network as the basic architecture for the $G_W(X)$ function that maps image points in the face-space. The architecture of convolutional nets is somewhat inspired by the structure of biological visual systems. Convolutional nets have been used successfully in a number of vision applications such as handwriting recognition (LeCun et al., 1989, 1998), and generic object recognition (LeCun et al., 2004). Several authors have advocated the use of Convolutional Networks for object detection (Vaillant et al., 1994; Nowlan and Platt, 1995; LeCun et al., 1998; Garcia and Delakis, 2002).

Convolutional networks are "end-to-end" trainable system that can operate on raw pixel images and learn low-level features and high-level representation in an integrated fashion. Each layer in a convolutional net is composed units organized in planes called feature maps. Each unit in a feature map takes inputs from a small neighborhood within the feature maps of the previous layer. Neighboring units in a feature map are connected to neighboring (possibly overlapping) windows. Each unit computes a weighted sum of its inputs and passes the result through a sigmoid saturation function. All units within a feature map share the same weights. Therefore, each feature map can be seen as convolving the feature maps of the previous layers with small-size kernels, and passing the sum of those convolutions through sigmoid functions. Units in a feature map detect local features at all locations on the previous layer.

Convolutional nets are advantageous because they can operate on raw images and can easily learn the type of shift-invariant local features that are relevant to image recognition. Furthermore, they are very efficient computationally for detection and recognition tasks involving a sliding window over large images (Vaillant et al., 1994; LeCun et al., 1998).

The network architecture used for training is shown in Figure 5. It is similar to LeNet5 (LeCun et al., 1998), but contains more feature maps. The network input is a 32×32 pixel gray-scale image. The first layer C1 is a convolutional layer with 8 feature maps of size 28×28. Each unit in each feature map is connected to a 5×5 neighborhood in the input. Contiguous units in C1 take input from neighborhood on the input that overlap by 4 pixels. The next layer, S2, is a so-called subsampling layer with 8 feature maps of size 14×14. Each unit in each map is connected to a 2×2 neighborhood in the corresponding feature map in C1. Contiguous units in S2 take input from contiguous, non-overlapping 2x2 neighborhoods in the corresponding map in C1. C3 is convolutional with 20 feature maps of size 10×10. Each unit in each feature map is connected to several 5×5 neighborhoods at identical locations in a subset of S2's feature maps. Different C3 maps take input from different subsets of S2 to break the symmetry and to force the maps to extract different features. S4 is a subsampling layer with 2×2 subsampling ratios containing 20 feature maps of size 5×5. Layer C5 is a convolutional layer with 120 feature maps of size 1×1 with 5×5 kernels. Each C5 map takes input from all 20 of S4's feature maps. The output layer has 9 outputs (since the face manifold is nine-dimensional) and is fully connected to C5 (such a full connection can be seen as a convolution with 1×1 kernels).

3.3 Training with a Contrastive Loss Function

The vector W contains all 63,493 weights and kernel coefficients in the convolutional network. They are all subject to training by minimizing a single loss function. A key element, and a novel contribution, of this paper is the design of the loss function.

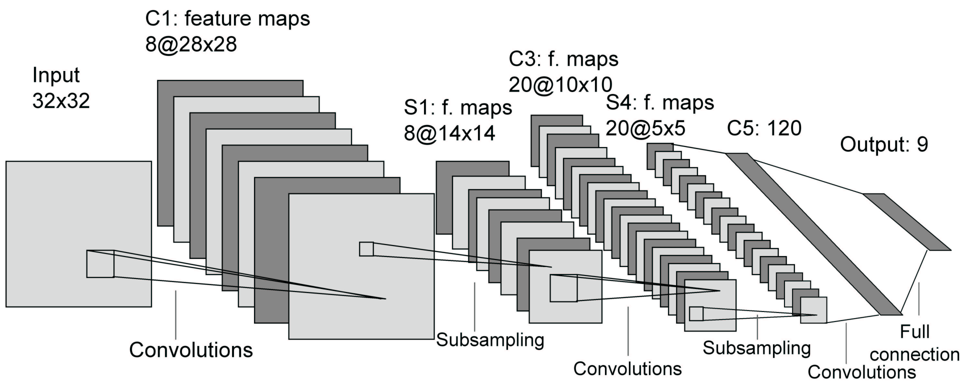


Figure 5: Architecture of convolutional network used for training. This represents one slice of the network with with a 32×32 input window. The slice includes all the elements that are necessary to compute a single output vector. The trained network is replicated over the full input image, producing one output vector for each 32×32 window stepped every 4 pixels horizontally and vertically. The process is repeated at multiple scales.

The training set $\mathcal{S}$ is composed of two subsets: the set $\mathcal{S}_1$ of training samples $(1, X^i, Z^i)$ containing a face annotated with the pose; and the set $\mathcal{S}_0$ of training sample $(0, X^i)$ containing a non-face image (background). The loss function $\mathcal{L}(W, \mathcal{S})$ is defined as the average over $\mathcal{S}_1$ of a per-sample loss function $L_1(W, Z^i, X^i)$, plus the average over $\mathcal{S}_0$ of a per-sample loss function $L_0(W, X^i)$:

$$\mathcal{L}(W, \mathcal{S}) = \frac{1}{|\mathcal{S}_1|} \sum_{i \in \mathcal{S}_1} L_1(W, Z^i, X^i) + \frac{1}{|\mathcal{S}_0|} \sum_{i \in \mathcal{S}_0} L_0(W, X^i). \tag{1}$$

Face samples whose pose is unknown can easily be accommodated by viewing Z as a "deterministic latent variable" over which the energy must be minimized. However, experiments reported in this paper only use training samples manually labeled with the pose.

For a particular positive training sample $(X^i, Z^i, 1)$, the per-sample loss L_1 should be designed in such a way that its minimization with respect to W will make the energy of the correct answer lower than the energies of all possible incorrect answers. Minimizing such a loss function will make the machine produce the right answer when running the energy-minimizing inference procedure. We can write this condition as:

Condition 1

$$E_W(Y^i = 1, Z^i, X^i) < E_W(Y, Z, X^i) \text{ for } Y \neq Y^i \text{ or } Z \neq Z^i.$$

Satisfying this condition can be done by satisfying the two following conditions

Condition 2

$$E_W(1, Z^i, X^i) < T \text{ and } E_W(1, Z^i, X^i) < \min_{Z \neq Z^i} E_W(1, Z, X^i).$$

Following LeCun and Huang (2005), we assume that the loss is a functional that depends on X only through the set of energies associated with X and all the possible values of Z and Y. This assumption allows us to decouple the design of the loss function from the internal structure (architecture) of the energy function. We also assume that there exist a W for which condition 2 is satisfied. This is a reasonable assumption, we merely ensure that the learning machine can produce the correct output for any single sample. We now show that, with our architecture, if we choose L_1 to be a strictly *monotonically increasing* function of $E_W(1,Z^i,X^i)$ (over the domain of E), then minimizing L_1 with respect to W will cause the machine to satisfy condition 2. The first inequality in 2 will obviously be satisfied by minimizing such a loss. The second inequality will be satisfied if $E_W(1,Z,X^i)$ has a single (non-degenerate) global minimum as a function of Z. The minimization of L_1 with respect to W will place this minimum at Z^i, and therefore will ensure that all other values of Z will have higher energy. Our energy function $E_W(1,Z,X) = ||G_W(X) - F(Z)||$ indeed has a single global minimum as a function of Z, because $F(Z)$ is injective and the norm is convex. The single global minimum is attained for $G_W(X) = F(Z)$. For our experiments, we simply chose:

$$L_1(W,1,Z,X) = E_W(1,Z,X)^2.$$

For a particular negative (non-face) training sample $(X^i,0)$, the per-sample loss L_0 should be designed in such a way that its minimization with respect to W will make the energy for $Y = 1$ and any value of Z higher than the energy for $Y = 0$ (which is equal to T). Minimizing such a loss function will make the machine produce the right answer when running the energy-minimizing inference procedure. We can write the condition for correct output as:

Condition 3

$$E_W(1,Z,X^i) > T \quad \forall Z$$

which can be re-written as:

Condition 4

$$E_W(1,\overline{Z},X^i) > T \quad \overline{Z} = \text{argmin}_z E_W(1,z,X^i).$$

Again, we assume that there exists a W for which condition 4 is satisfied. It is easy to see that, with our architecture, if we choose L_0 to be a strictly *monotonically decreasing* function of $E_W(1,\overline{Z},X^i)$ (over the domain of E), then minimizing L_0 with respect to W will cause the machine to satisfy condition 4. For our experiments, we simply chose:

$$L_0(W,0,X^i) = K\exp[-E(1,\overline{Z},X^i)]$$

where K is a positive constant. A nice property of this function is that it is bounded below by 0, and that its gradient vanishes we approach the minimum.

The entire system was trained by minimizing average value of the loss function in Eq. (1) with respect to the parameter W. We used a stochastic version of the Levenberg-Marquardt algorithm with diagonal approximation of the Hessian (LeCun et al., 1998).

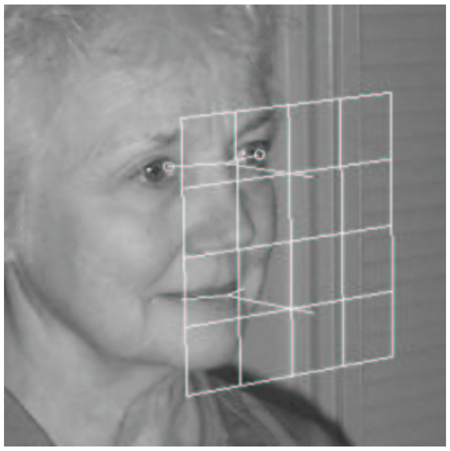


Figure 6: Screenshot from annotation tool.

3.4 Running the Machine

The detection system operates on raw grayscale images. The convolutional network is applied to all 32×32 sub-windows of the image, stepped every 4 pixels horizontally and vertically. Because the layers are convolutional, applying two replicas of the network in Figure 5 to two overlapping input windows leads to a considerable amount of redundant computation. Eliminating the redundant computation yields a dramatic speed up: each layer of the convolutional network is extended so as to cover the entire input image. The output is also replicated the same way. Due to the two 2×2 subsampling layers, we obtain one output vector every 4×4 pixels.

To detect faces in a size-invariant fashion, the network is applied to multiple down-scaled versions of the image over a range of scales stepped by a factor of $\sqrt{2}$. At each scale and location, the network's 9-dimensional output vector is compared to the closest point on the face manifold (whose position indicates the pose of the candidate face). The system collects a list of all locations and scales of output vectors closer to the face manifold than the detection threshold. After examining all scales, the system identifies groups of overlapping detections in the list and discards all but the strongest (closest to the manifold) from each group within an exclusion area of a preset size. No attempt is made to combine detections or apply any voting scheme.

4. Experiments and Results

Using the architecture described in Section 3, we built a detector to locate faces and estimate two pose parameters: yaw from left to right profile, and in-plane rotation from -45 to 45 degrees. The machine was trained to be robust against pitch variation.

In this section, we first describe the training protocol for this network, and then give the results of two sets of experiments. The first set of experiments tests whether training for the two tasks together improves performance on both. The second set allows comparisons between our system and other published multi-view detectors.

4.1 Training

The images we used for training were collected at NEC Labs. All face images were manually annotated with appropriate poses. The annotation process was greatly simplified by using a simple tool for specifying a location and approximate pose of a face. The user interface for this tool is shown in Figure 6. Annotation process is done by first clicking on the midpoint between the eyes and on the center of the mouth. The tool then draws a perspective grid in front of the face and the user adjusts it to be parallel to the face plane. This process yields estimates for all six pose parameters: location (x,y), three angles (yaw, pitch, and roll) and scale. The images were annotated in such a way that the midpoint between the eyes and on the center of the mouth are positioned in the center of the image. This allows these two points to stay fixed when the pose changes from left to right profile. The downside is that profiles occupy only half of the image.

In this manner we annotated about 30,000 faces in images from various sources. Each face was then cropped and scaled so that the eye midpoint and the mouth midpoint appeared in canonical positions, 10 pixels apart, in 32×32-pixel image with some moderate variation of location and scale. The resulting images where mirrored horizontally, to yield roughly 60,000 faces. We removed some portion of faces from this set to yield a roughly uniform distribution of poses from left profile to right profile. Unfortunately, the amount of variation in pitch (up/down) was not sufficient to do the same. This was the reason for training our system to be robust against pitch variation instead of estimating the pitch angle. The roll variation was added by randomly rotating the images in the range of $[-45,45]$ degrees. The resulting training set consisted of $52,850$, 32x32 grey-level images of faces with uniform distribution of poses.

The initial set of negative training samples consisted of $52,850$ image patches chosen randomly from non-face areas in a variety of images. For the second set of tests, half of these images were replaced with image patches obtained by running the initial version of the detector on the training images and collecting false detections.

Each training image was used 5 times during training, with random variations in scale (from x$\sqrt{2}$ to x$(1+\sqrt{2})$), in-plane rotation ($\pm 45°$), brightness (± 20), and contrast (from 0.8 to 1.3).

To train the network, we made 9 passes through this data, though it mostly converged after about the first 6 passes. The training system was implemented in the Lush language (Bottou and LeCun, 2002). The total training time was about 26 hours on a 2Ghz Pentium 4. At the end of training, the network had converged to an equal error rate of 5% on the training data and 6% on a separate test set of 90,000 images.

A standalone version of the detection system was implemented in the C language. It can detect, locate, and estimate the pose of faces that are between 40 and 250 pixels high in a 640×480 image at roughly 5 frames per second on a 2.4GHz Pentium 4.

4.2 Synergy Tests

The goal of the synergy test was to verify that both face detection and pose estimation benefit from learning and running in parallel. To test this claim we built three networks with almost identical architectures, but trained to perform different tasks. The first one was trained for simultaneous face detection and pose estimation (combined), the second was trained for detection only and the third for pose estimation only. The "detection only" network had only one output for indicating whether or not its input was a face. The "pose only" network was identical to the combined network, but trained on faces only (no negative examples). Figure 7 shows the results of running these networks

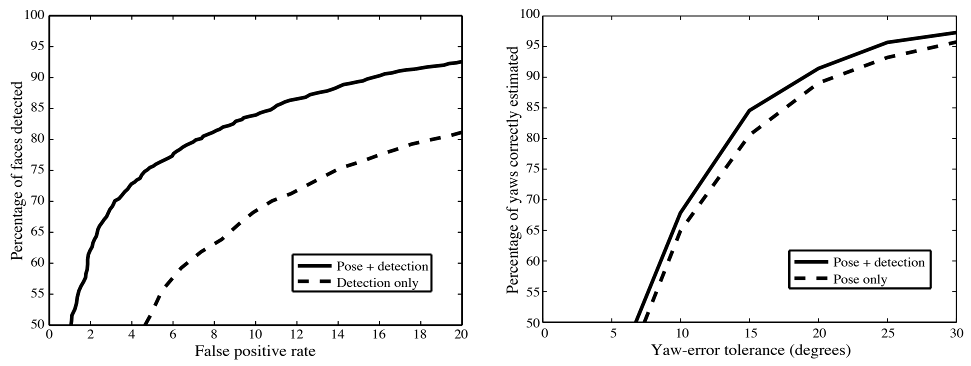


Figure 7: Synergy test. Left: ROC curves for the pose-plus-detection and detection-only networks. (The x axis is the false positive rate per image). Right: frequency with which the pose-plus-detection and pose-only networks correctly estimated the yaws within various error tolerances.

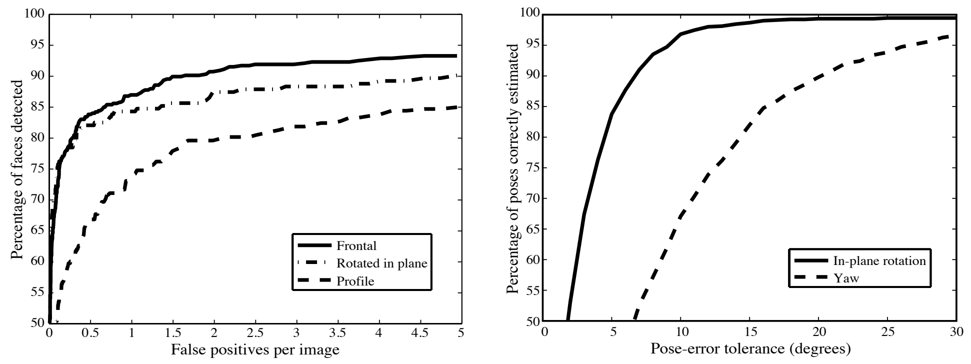


Figure 8: Results on standard data sets. Left: ROC curves for our detector on the three data sets. The x axis is the average number of false positives per image over all three sets, so each point corresponds to a single detection threshold. Right: frequency with which yaw and roll are estimated within various error tolerances.

on our 10,000 test images. In both these graphs, we see that the pose-plus-detection network had better performance, confirming that training for each task benefits the other.

4.3 Standard Data Sets

There is no standard data set that spans the range of poses our system is designed to handle. There are, however, data sets that have been used to test more restricted face detectors, each set focusing on a particular variation in pose. By testing a single detector with all of these sets, we can compare our performance against published systems. As far as we know, we are the first to publish results for a single detector on all these data sets. The details of these sets are described below:

• MIT+CMU (Sung and Poggio, 1998; Rowley et al., 1998a) – 130 images for testing frontal face detectors. We count 517 faces in this set, but the standard tests only use a subset of 507 faces, because 10 faces are in the wrong pose or otherwise not suitable for the test. (Note: about 2% of the faces in the standard subset are badly-drawn cartoons, which we do not intend our system to detect. Nevertheless, we include them in the results we report.)

• TILTED (Rowley et al., 1998b) – 50 images of frontal faces with in-plane rotations. 223 faces out of 225 are in the standard subset. (Note: about 20% of the faces in the standard subset are outside of the $\pm 45^\circ$ rotation range for which our system is designed. Again, we still include these in our results.)

• PROFILE (Schneiderman and Kanade, 2000) – 208 images of faces in profile. There seems to be some disagreement about the number of faces in the standard set of annotations: Schneiderman and Kanade (2000) reports using 347 faces of the 462 that we found, Jones and Viola (2003) reports using 355, and we found 353 annotations. However, these discrepancies should not significantly effect the reported results.

We counted a face as being detected if 1) at least one detection lay within a circle centered on the midpoint between the eyes, with a radius equal to 1.25 times the distance from that point to the midpoint of the mouth, and 2) that detection came at a scale within a factor of two of the correct scale for the face's size. We counted a detection as a false positive if it did not lie within this range for any of the faces in the image, including those faces not in the standard subset.

The left graph in Figure 8 shows ROC curves for our detector on the three data sets. Figures 9, 10 show detection results on various poses. Table 1 shows our detection rates compared against other multi-view systems for which results were given on these data sets. We want to stress here that all these systems are tested in a pose specific manner: for example, a detector tested on TILTED set is trained only on frontal tilted faces. Such a detector will not be able to detect non frontal tilted faces. Combining all pose variations in one system obviously will increase the number of false positives, since false positives of view-based detectors are not necessarily correlated. Our system is designed to handle all pose variations. This makes the comparison in Table 1 somewhat unfair to our system, but we don't see any other way of comparison against other systems.

The table shows that our results on the TILTED and PROFILE sets are similar to those of the two Jones & Viola detectors, and even approach those of the Rowley *et al* and Schneiderman & Kanade non-real-time detectors. Those detectors, however, are not designed to handle all variations in pose, and do not yield pose estimates. More recent system reported in Huang et al. (2004) is also real-time and can handle all pose variation, but doesn't yield pose estimates. Unfortunately, they also report the results of pose specific detectors. These results are not shown in Table 1, because they report different points on ROC curve in the PROFILE experiment (86.2% for 0.42 f.p per image) and they didn't test on the TILTED set. Even though they trained a combined detector for all pose variations, they did not test it the way we did. Their test consists in running the full detector on the PROFILE set rotated by [-30, 30] degrees in-plane. Unfortunately, they do not provide enough details to recreate their test set.

The right side of Figure 8 shows our performance at pose estimation. To make this graph, we fixed the detection threshold at a value that resulted in about 0.5 false positives per image over all three data sets. We then compared the pose estimates for all detected faces (including those not in the standard subsets) against our manual pose annotations. Note that this test is more difficult than typical tests of pose estimation systems, where faces are first localized by hand. When we hand-localize these faces, 89% of yaws and 100% of in-plane rotations are correctly estimated to within 15°.

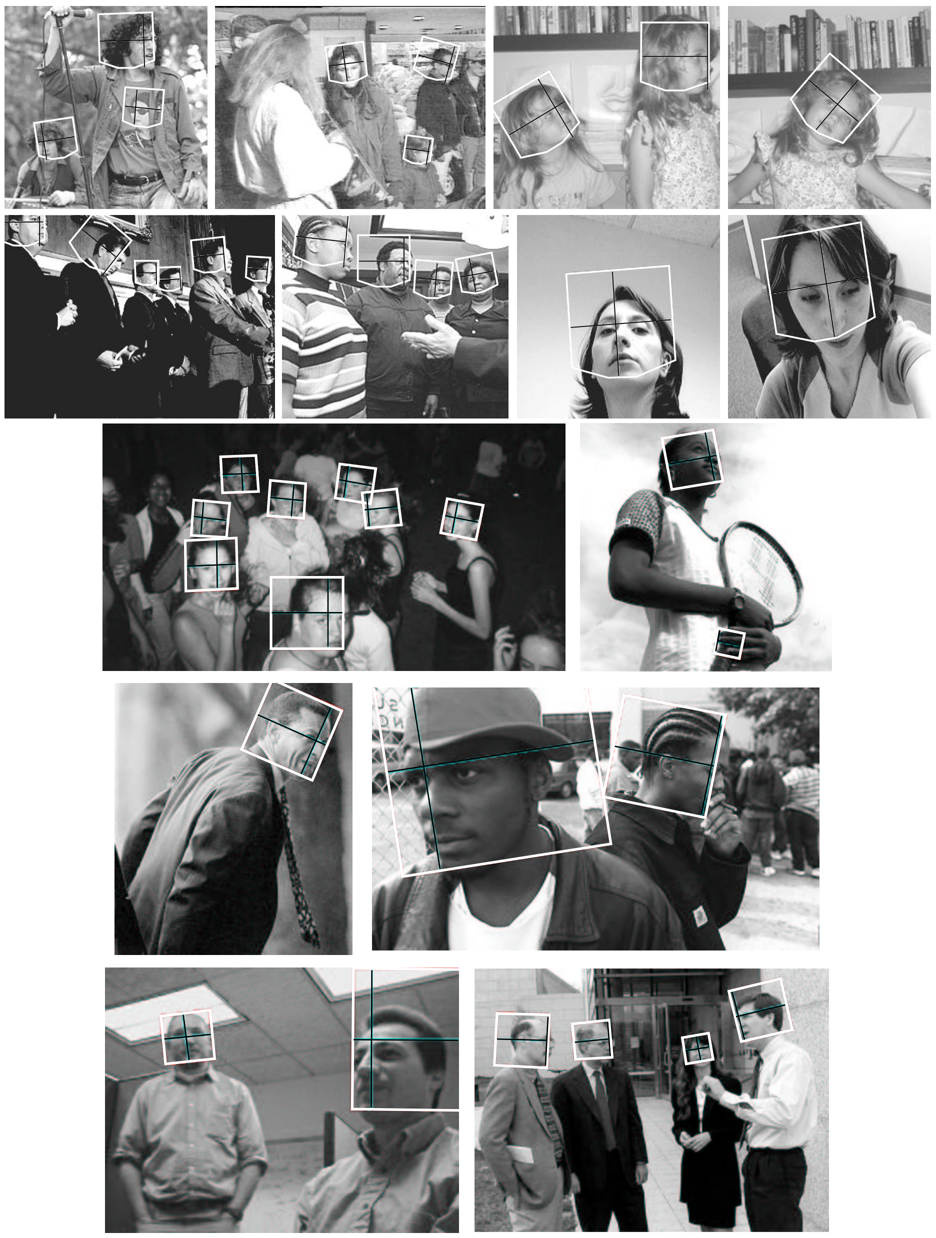


Figure 9: Some example face detections. Each white box shows the location of a detected face. The angle of each box indicates the estimated in-plane rotation. The black crosshairs within each box indicate the estimated yaw.

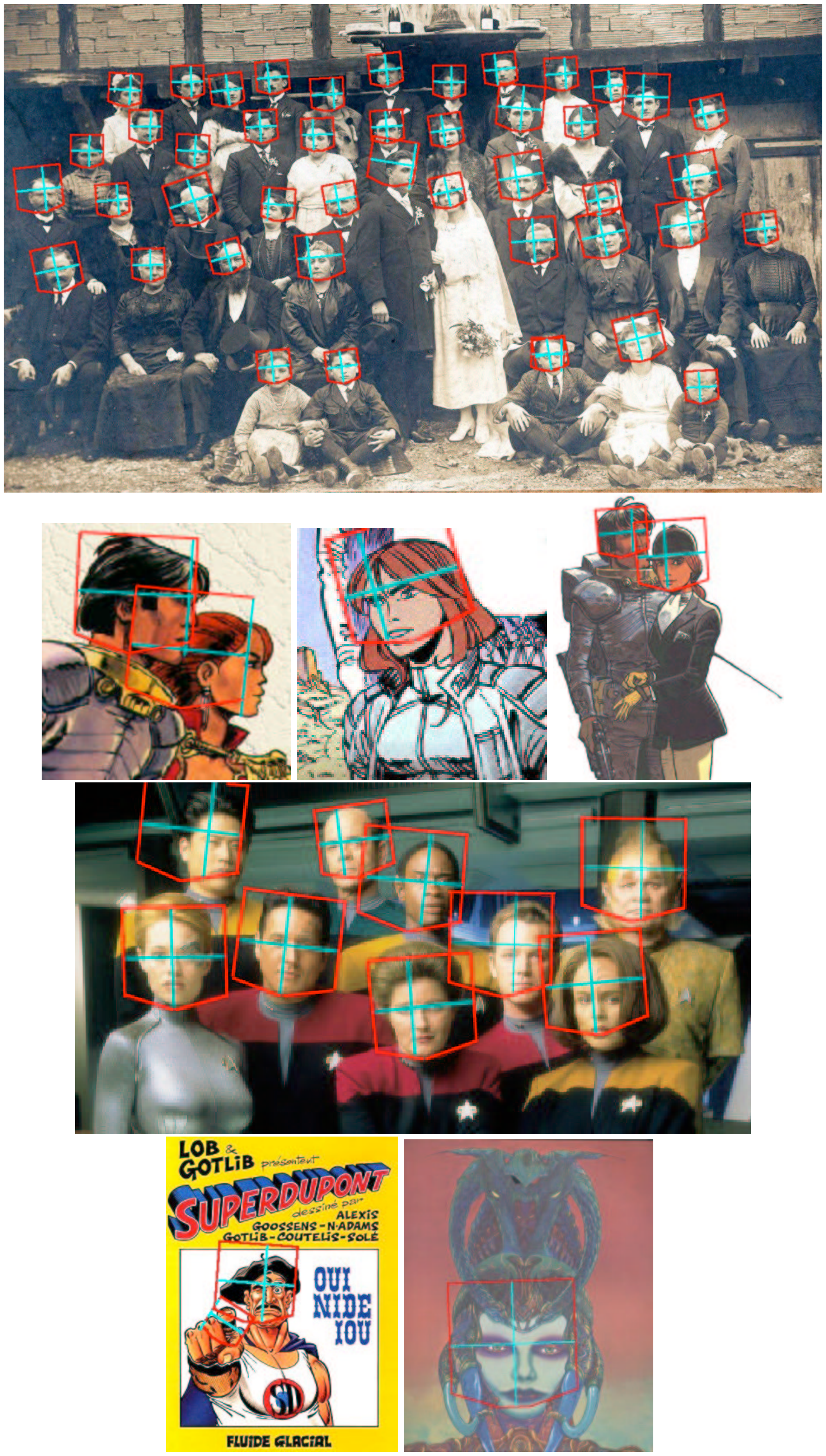


Figure 10: More examples of face detections.

Data set →	TILTED		PROFILE		MIT+CMU	
False positives per image →	4.42	26.90	.44	3.36	.50	1.28
Our detector	**90%**	**97%**	**67%**	**83%**	**83%**	**88%**
Jones and Viola (2003) (tilted)	**90%**	**95%**	x		x	
Jones and Viola (2003) (profile)	x		**70%**	**83%**	x	
Rowley et al. (1998a)	89%	96%	x		x	
Schneiderman and Kanade (2000)	x		86%	93%	x	

Table 1: Comparisons of our results with other multi-view detectors. Each column shows the detection rates for a given average number of false positives per image (these rates correspond to those for which other authors have reported results). Results for real-time detectors are shown in bold. Note that ours is the only single detector that can be tested on all data sets simultaneously.

5. Conclusion

The system we have presented here integrates detection and pose estimation by training a convolutional network to map faces to points on a manifold, parameterized by pose, and non-faces to points far from the manifold. The network is trained by optimizing a loss function of three variables—image, pose, and face/non-face label. When the three variables match, the energy function is trained to have a small value, when they do not match, it is trained to have a large value.

This system has several desirable properties:

• The use of a convolutional network makes it fast. At typical webcam resolutions, it can process 5 frames per second on a 2.4Ghz Pentium 4.

• It is robust to a wide range of poses, including variations in yaw up to $\pm 90^\circ$, in-plane rotation up to $\pm 45^\circ$, and pitch up to $\pm 60^\circ$. This has been verified with tests on three standard data sets, each designed to test robustness against a single dimension of pose variation.

• At the same time that it detects faces, it produces estimates of their pose. On the standard data sets, the estimates of yaw and in-plane rotation are within 15° of manual estimates over 80% and 95% of the time, respectively.

We have shown experimentally that our system's accuracy at *both* pose estimation and face detection is increased by training for the two tasks together.

References

L. Bottou and Y. LeCun. *The Lush Manual.* `http://lush.sf.net`, 2002.

R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

F. Fleuret and D. Geman. Coarse-to-fine face detection. *IJCV*, pages 85–107, 2001.

C. Garcia and M. Delakis. A neural architecture for fast and robust face detection. *IEEE-IAPR Int. Conference on Pattern Recognition*, pages 40–43, 2002.

C. Huang, B. Wu, H. Ai, and S. Lao. Omni-directional face detection based on real adaboost. In *International Conference on Image Processing*, Singapore, 2004.

M. Jones and P. Viola. Fast multi-view face detection. Technical Report TR2003-96, Mitsubishi Electric Research Laboratories, 2003.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, Winter 1989.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

Y. LeCun, R. Hadsell, S. Chopra, F.-J. Huang, and M.-A. Ranzato. A tutorial on energy-based learning. In *Predicting Structured Outputs*. Bakir et al. (eds),MIT Press, 2006.

Y. LeCun and F. J. Huang. Loss functions for discriminative training of energy-based models. In *Proc. of the 10-th International Workshop on Artificial Intelligence and Statistics (AIStats'05)*, 2005.

Y. LeCun, F.-J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of CVPR'04*. IEEE Press, 2004.

S. Z. Li, L. Zhu, Z. Zhang, A. Blake, H. Zhang, and H. Shum. Statistical learning of multi-view face detection. In *Proceedings of the 7th European Conference on Computer Vision-Part IV*, 2002.

Y. Li, S. Gong, and H. Liddell. Support vector regression and classification based multi-view face detection and recognition. In *Face and Gesture*, 2000.

S. Nowlan and J. Platt. A convolutional neural network hand tracker. In *Advances in Neural Information Processing Systems (NIPS 1995)*, pages 901–908, San Mateo, CA, 1995. Morgan Kaufmann.

M. Osadchy, M. Miller, and Y. LeCun. Synergistic face detection and pose estimation with energy-based model. In *Advances in Neural Information Processing Systems (NIPS 2004)*. MIT Press, 2005.

A. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces for face recognition. In *CVPR*, 1994.

H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *PAMI*, 20:22–38, 1998a.

H. A. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. In *Computer Vision and Pattern Recognition*, 1998b.

H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. In *Computer Vision and Pattern Recognition*, 2000.

K. Sung and T. Poggio. Example-based learning of view-based human face detection. *PAMI*, 20: 39–51, 1998.

R. Vaillant, C. Monrocq, and Y. LeCun. Original approach for the localisation of objects in images. *IEE Proc on Vision, Image, and Signal Processing*, 141(4):245–250, August 1994.

P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 511–518, 2001.

M.-H. Yang, D. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *PAMI*, 24(1):34–58, 2002.

Journal of Machine Learning Research 8 (2007) 1217-1260 Submitted 4/06; Revised 4/07; Published 6/07

Maximum Entropy Density Estimation with Generalized Regularization and an Application to Species Distribution Modeling

Miroslav Dudík MDUDIK@CS.PRINCETON.EDU
Princeton University
Department of Computer Science
35 Olden Street
Princeton, NJ 08540

Steven J. Phillips PHILLIPS@RESEARCH.ATT.COM
AT&T Labs − Research
180 Park Avenue
Florham Park, NJ 07932

Robert E. Schapire SCHAPIRE@CS.PRINCETON.EDU
Princeton University
Department of Computer Science
35 Olden Street
Princeton, NJ 08540

Editor: John Lafferty

Abstract

We present a unified and complete account of maximum entropy density estimation subject to constraints represented by convex potential functions or, alternatively, by convex regularization. We provide fully general performance guarantees and an algorithm with a complete convergence proof. As special cases, we easily derive performance guarantees for many known regularization types, including ℓ_1, ℓ_2, ℓ_2^2, and $\ell_1+\ell_2^2$ style regularization. We propose an algorithm solving a large and general subclass of generalized maximum entropy problems, including all discussed in the paper, and prove its convergence. Our approach generalizes and unifies techniques based on information geometry and Bregman divergences as well as those based more directly on compactness. Our work is motivated by a novel application of maximum entropy to species distribution modeling, an important problem in conservation biology and ecology. In a set of experiments on real-world data, we demonstrate the utility of maximum entropy in this setting. We explore effects of different feature types, sample sizes, and regularization levels on the performance of maxent, and discuss interpretability of the resulting models.

Keywords: maximum entropy, density estimation, regularization, iterative scaling, species distribution modeling

1. Introduction

The maximum entropy (maxent) approach to density estimation was first proposed by Jaynes (1957), and has since been used in many areas of computer science and statistical learning, especially natural language processing (Berger et al., 1996; Della Pietra et al., 1997). In maxent, one is given a set of samples from a target distribution over some space, and a set of known constraints on the distribution. The distribution is then estimated by a distribution of maximum entropy satisfying

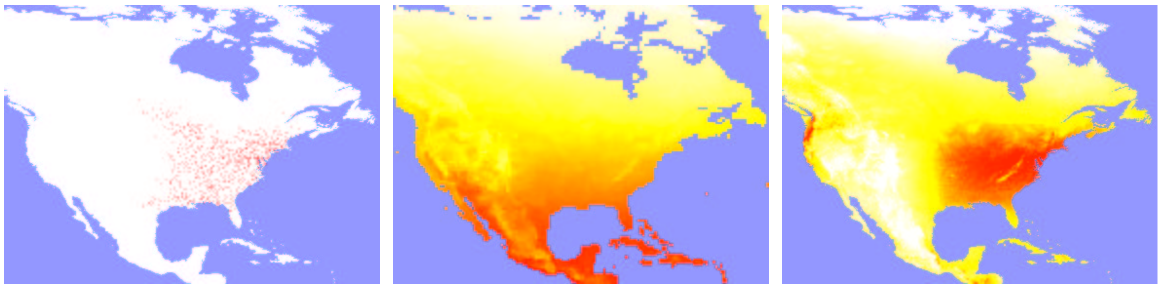


Figure 1: Left to right: Yellow-throated Vireo training localities from the first random partition, an example environmental variable (annual average temperature, higher values in red), maxent prediction using linear, quadratic and product features. Prediction strength is shown as white (weakest) to red (strongest); reds could be interpreted as suitable conditions for the species.

the given constraints. The constraints are often represented using a set of *features* (real-valued functions) on the space, with the expectation of every feature required to match its empirical average. By convex duality, this turns out to be the unique Gibbs distribution maximizing the likelihood of the samples, or, equivalently, minimizing the empirical log loss. (Maxent and its dual are described more rigorously in Section 2.)

The work in this paper was motivated by a new application of maxent to the problem of modeling the distribution of a plant or animal species, a critical problem in conservation biology. Input data for species distribution modeling consists of occurrence locations of a particular species in a region and of environmental variables for that region. Environmental variables may include topographical layers, such as elevation and aspect, meteorological layers, such as annual precipitation and average temperature, as well as categorical layers, such as vegetation and soil type. Occurrence locations are commonly derived from specimen collections in natural history museums and herbaria. In the context of maxent, occurrences correspond to samples, the map divided into a finite number of cells is the sample space, and environmental variables or functions derived from them are features (see Figure 1 for an example). The number of occurrences for individual species is frequently quite small by machine learning standards, for example, a hundred or less.

It should not be surprising that maxent can severely overfit training data when the constraints on the output distribution are based on empirical averages, as described above, especially if there is a very large number of features. For instance, in our application, we sometimes consider threshold features for each environmental variable. These are binary features equal to one if an environmental variable is larger than a fixed threshold and zero otherwise. Thus, there is a continuum of features for each variable, and together they force the output distribution to be non-zero only at values achieved by the samples. The problem is that in general, the empirical averages of features will almost never be equal to their true expectations, so the target distribution itself does not satisfy the constraints imposed on the output distribution. From the dual perspective, the family of Gibbs distributions is too expressive and the algorithm overfits. Common approaches to counter overfitting are parameter regularization (Lau, 1994; Chen and Rosenfeld, 2000; Lebanon and Lafferty, 2001; Zhang, 2005), introduction of a prior (Williams, 1995; Goodman, 2004), feature selection (Berger et al., 1996; Della Pietra et al., 1997), discounting (Lau, 1994; Rosenfeld, 1996; Chen and Rosenfeld, 2000) and constraint relaxation (Khudanpur, 1995; Kazama and Tsujii, 2003; Jedynak and Khudanpur, 2005). Thus, there are many ways of modifying maxent to control overfitting calling for a general treatment.

In this work, we study a generalized form of maxent. Although mentioned by other authors as *fuzzy maxent* (Lau, 1994; Chen and Rosenfeld, 2000; Lebanon and Lafferty, 2001), we give the first complete theoretical treatment of this very general framework, including fully general and unified performance guarantees, algorithms, and convergence proofs. Independently, Altun and Smola (2006) derive a different theoretical treatment (see discussion below).

As special cases, our results allow us to easily derive performance guarantees for many known regularized formulations, including ℓ_1, ℓ_2, ℓ_2^2, and $\ell_1 + \ell_2^2$ regularizations. More specifically, we derive guarantees on the performance of maxent solutions compared to the "best" Gibbs distribution $q^\star$ defined by a weight vector $\boldsymbol{\lambda}^\star$. Our guarantees are derived by bounding deviations of empirical feature averages from their expectations, a setting in which we can take advantage of a wide array of uniform convergence results. For example, for a finite set of features bounded in $[0,1]$, we can use Hoeffding's inequality and the union bound to show that the true log loss of the ℓ_1-regularized maxent solution will be with high probability worse by no more than an additive $O(\|\boldsymbol{\lambda}^\star\|_1 \sqrt{(\ln n)/m})$ compared with the log loss of the Gibbs distribution $q^\star$, where n is the number of features and m is the number of samples. For an infinite set of binary features with VC-dimension d, the difference between the ℓ_1-regularized maxent solution and $q^\star$ is at most $O(\|\boldsymbol{\lambda}^\star\|_1 \sqrt{d \ln(m^2/d)/m})$. Note that these bounds drop quickly with an increasing number of samples and depend only moderately on the number or complexity of the features, even admitting an extremely large number of features from a class of bounded VC-dimension. For maxent with ℓ_2 and ℓ_2^2-style regularization, it is possible to obtain bounds which are independent of the number of features, provided that the feature vector can be bounded in the ℓ_2 norm.

In the second part, we propose algorithms solving a large and general subclass of generalized maxent problems. We show convergence of our algorithms using a technique that unifies previous approaches and extends them to a more general setting. Specifically, our unified approach generalizes techniques based on information geometry and Bregman divergences (Della Pietra et al., 1997, 2001; Collins et al., 2002) as well as those based more directly on compactness. The main novel ingredient is a modified definition of an auxiliary function, a customary measure of progress, which we view as a surrogate for the difference between the primal and dual objective rather than a bound on the change in the dual objective.

Standard maxent algorithms such as iterative scaling (Darroch and Ratcliff, 1972; Della Pietra et al., 1997), gradient descent, Newton and quasi-Newton methods (Cesa-Bianchi et al., 1994; Malouf, 2002; Salakhutdinov et al., 2003), and their regularized versions (Lau, 1994; Williams, 1995; Chen and Rosenfeld, 2000; Kazama and Tsujii, 2003; Goodman, 2004; Krishnapuram et al., 2005) perform a sequence of feature weight updates until convergence. In each step, they update all feature weights. This is impractical when the number of features is very large. Instead, we propose a sequential update algorithm that updates only one feature weight in each iteration, along the lines of algorithms studied by Collins, Schapire, and Singer (2002), and Lebanon and Lafferty (2001). This leads to a boosting-like approach permitting the selection of the best feature from a very large class. For instance, for ℓ_1-regularized maxent, the best threshold feature associated with a single variable can be found in a single linear pass through the (pre-sorted) data, even though conceptually we are selecting from an infinite class of features. Other boosting-like approaches to density estimation have been proposed by Welling, Zemel, and Hinton (2003), and Rosset and Segal (2003).

For cases when the number of features is relatively small, yet we want to use benefits of regularization to prevent overfitting on small sample sets, it might be more efficient to solve generalized maxent by parallel updates. In Section 7, we give a parallel-update version of our algorithm with a proof of convergence.

In the last section, we return to species distribution modeling, and use it as a setting to test our ideas. In particular, we apply ℓ_1-regularized maxent to estimate distributions of bird species in North America. We present learning curves for several different feature classes derived for four species with a varying number of occurrence records. We also explore effects of regularization on the test log loss and interpretability of the resulting models. A more comprehensive set of experiments is evaluated by Phillips, Dudík, and Schapire (2004). The biological application is explored in more detail by Phillips, Anderson, and Schapire (2006).

1.1 Previous Work

There have been many studies of maxent and logistic regression, which is a conditional version of maxent, with ℓ_1-style regularization (Khudanpur, 1995; Williams, 1995; Kazama and Tsujii, 2003; Ng, 2004; Goodman, 2004; Krishnapuram et al., 2005), ℓ_2^2-style regularization (Lau, 1994; Chen and Rosenfeld, 2000; Lebanon and Lafferty, 2001; Zhang, 2005) as well as some other types of regularization such as $\ell_1 + \ell_2^2$-style (Kazama and Tsujii, 2003), ℓ_2-style regularization (Newman, 1977) and a smoothed version of ℓ_1-style regularization (Dekel et al., 2003). In a recent work, Altun and Smola (2006) derive duality and performance guarantees for settings in which the entropy is replaced by an arbitrary Bregman or Csiszár divergence and regularization takes the form of a norm raised to a power greater than one. With the exception of Altun and Smola's work and Zhang's work, the previous studies do not give performance guarantees applicable to our case, although Krishnapuram et al. (2005) and Ng (2004) prove guarantees for ℓ_1-regularized logistic regression. Ng also shows that ℓ_1-regularized logistic regression may be superior to the ℓ_2^2-regularized version in a scenario when the number of features is large and only a small number of them is relevant. Our results indicate a similar behavior for unconditional maxent.

In the context of linear models, ℓ_2^2, ℓ_1, and $\ell_1 + \ell_2^2$ regularization have been used under the names *ridge regression* (Hoerl and Kennard, 1970), *lasso regression* (Tibshirani, 1996), and *elastic nets* (Zou and Hastie, 2005). Lasso regression, in particular, has provoked a lot of interest in recent statistical theory and practice. The frequently mentioned benefit of the lasso is its bias toward sparse solutions. The same bias is present also in ℓ_1-regularized maxent, but we do not analyze this bias in detail. Our interest is in deriving performance guarantees. Similar guarantees were derived by Donoho and Johnstone (1994) for linear models with the lasso penalty. The relationship between the lasso approximation and the sparsest approximation is explored, for example, by Donoho and Elad (2003).

Quite a number of approaches have been suggested for species distribution modeling, including neural nets, nearest neighbors, genetic algorithms, generalized linear models, generalized additive models, bioclimatic envelopes, boosted regression trees, and more; see Elith (2002) and Elith et al. (2006) for a comprehensive comparison. The latter work evaluates ℓ_1-regularized maxent as one of a group of twelve methods in the task of modeling species distributions. Maxent is among the best methods alongside boosted decision trees (Schapire, 2002; Leathwick et al., 2006), generalized dissimilarity models (Ferrier et al., 2002) and multivariate adaptive regression splines with the community level selection of basis functions (Moisen and Frescino, 2002; Leathwick et al., 2005).

Among these, however, maxent is the only method designed for presence-only data. It comes with a statistical interpretation that allows principled extensions, for example, to cases where the sampling process is biased (Dudík et al., 2005).

2. Preliminaries

Our goal is to estimate an unknown density π over a *sample space* $\mathcal{X}$ which, for the purposes of this paper, we assume to be finite.[1] As empirical information, we are typically given a set of *samples* $x_1, \ldots, x_m$ drawn independently at random according to π. The corresponding empirical distribution is denoted by $\tilde{\pi}$:

$$\tilde{\pi}(x) = \frac{|\{1 \le i \le m : x_i = x\}|}{m} \ .$$

We also are given a set of *features* $f_1, \ldots, f_n$ where $f_j : \mathcal{X} \to \mathbb{R}$. The vector of all n features is denoted by $\mathbf{f}$ and the image of $\mathcal{X}$ under $\mathbf{f}$, the *feature space*, is denoted by $\mathbf{f}(\mathcal{X})$. For a distribution π and function f, we write $\pi[f]$ to denote the expected value of f under distribution π:

$$\pi[f] = \textstyle\sum_{x \in \mathcal{X}} \pi(x) f(x) \ .$$

In general, $\tilde{\pi}$ may be quite distant, under any reasonable measure, from π. On the other hand, for a given function f, we do expect $\tilde{\pi}[f]$, the empirical average of f, to be rather close to its true expectation $\pi[f]$. It is quite natural, therefore, to seek an approximation p under which f_j's expectation is equal to $\tilde{\pi}[f_j]$ for every f_j. There will typically be many distributions satisfying these constraints. The *maximum entropy principle* suggests that, from among all distributions satisfying these constraints, we choose the one of maximum entropy, that is, the one that is closest to uniform. Here, as usual, the entropy of a distribution p on $\mathcal{X}$ is defined to be $\mathrm{H}(p) = -\sum_{x \in \mathcal{X}} p(x) \ln p(x)$.

However, the *default estimate* of π, that is, the distribution we would choose if we had no sample data, may be in some cases non-uniform. In a more general setup, we therefore seek a distribution that minimizes entropy relative to the default estimate q_0. The relative entropy, or Kullback-Leibler divergence, is an information theoretic measure defined as

$$\mathrm{D}(p \parallel q) = p[\ln(p/q)] \ .$$

Minimizing entropy relative to q_0 corresponds to choosing a distribution that is closest to q_0. When q_0 is uniform then minimizing entropy relative to q_0 is equivalent to maximizing entropy.

Instead of minimizing entropy relative to q_0, we can consider all *Gibbs distributions* of the form

$$q_{\boldsymbol{\lambda}}(x) = \frac{q_0(x) e^{\boldsymbol{\lambda} \cdot \mathbf{f}(x)}}{Z_{\boldsymbol{\lambda}}}$$

where $Z_{\boldsymbol{\lambda}} = \sum_{x \in \mathcal{X}} q_0(x) e^{\boldsymbol{\lambda} \cdot \mathbf{f}(x)}$ is a normalizing constant, and $\boldsymbol{\lambda} \in \mathbb{R}^n$. It can be proved (Della Pietra et al., 1997) that the maxent distribution is the same as the maximum likelihood distribution from the closure of the set of Gibbs distributions, that is, the distribution q that achieves the supremum of $\prod_{i=1}^{m} q_{\boldsymbol{\lambda}}(x_i)$ over all values of $\boldsymbol{\lambda}$, or equivalently, the infimum of the empirical log loss (negative normalized log likelihood)

$$\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) = -\frac{1}{m} \sum_{i=1}^{m} \ln q_{\boldsymbol{\lambda}}(x_i) \ .$$

1. In this paper, we are concerned with densities relative to the counting measure on $\mathcal{X}$. These correspond to probability mass functions.

The convex programs corresponding to the two optimization problems are

$$\min_{p\in\Delta} \mathrm{D}(p \parallel q_0) \text{ subject to } p[\mathbf{f}] = \tilde{\pi}[\mathbf{f}] \ , \tag{1}$$

$$\inf_{\boldsymbol{\lambda}\in\mathbb{R}^n} \mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) \tag{2}$$

where Δ is the simplex of probability distributions over $\mathcal{X}$.

In general, we use

$$\mathrm{L}_r(\boldsymbol{\lambda}) = -r[\ln q_{\boldsymbol{\lambda}}]$$

to denote the log loss of $q_{\boldsymbol{\lambda}}$ relative to the distribution r. It differs from relative entropy $\mathrm{D}(r \parallel q_{\boldsymbol{\lambda}})$ only by the constant $\mathrm{H}(r)$. We will use the two interchangeably as objective functions.

3. Convex Analysis Background

Throughout this paper we make use of convex analysis. The necessary background is provided in this section. For a more detailed exposition see for example Rockafellar (1970), or Boyd and Vandenberghe (2004).

Consider a function $\psi : \mathbb{R}^n \to (-\infty, \infty]$. The *effective domain* of ψ is the set $\operatorname{dom}\psi = \{\mathbf{u} \in \mathbb{R}^n : \psi(\mathbf{u}) < \infty\}$. A point $\mathbf{u}$ where $\psi(\mathbf{u}) < \infty$ is called *feasible*. The *epigraph* of ψ is the set of points above its graph $\{(\mathbf{u},t) \in \mathbb{R}^n \times \mathbb{R} : t \geq \psi(\mathbf{u})\}$. We say that ψ is *convex* if its epigraph is a convex set. A convex function is called *proper* if it is not uniformly equal to ∞. It is called *closed* if its epigraph is closed. For a proper convex function, closedness is equivalent to lower semi-continuity (ψ is lower semi-continuous if $\liminf_{\mathbf{u}'\to\mathbf{u}} \psi(\mathbf{u}') \geq \psi(\mathbf{u})$ for all $\mathbf{u}$).

If ψ is a closed proper convex function then its *conjugate* $\psi^* : \mathbb{R}^n \to (-\infty, \infty]$ is defined by

$$\psi^*(\boldsymbol{\lambda}) = \sup_{\mathbf{u}\in\mathbb{R}^n} [\boldsymbol{\lambda}\cdot\mathbf{u} - \psi(\mathbf{u})] \ .$$

The conjugate provides an alternative description of ψ in terms of tangents of ψ's epigraph. The definition of the conjugate immediately yields *Fenchel's inequality*

$$\forall \boldsymbol{\lambda}, \mathbf{u} : \ \boldsymbol{\lambda}\cdot\mathbf{u} \leq \psi^*(\boldsymbol{\lambda}) + \psi(\mathbf{u}) \ .$$

In fact, $\psi^*(\boldsymbol{\lambda})$ is defined to give the tightest bound of the form above. It turns out that ψ^* is also a closed proper convex function and $\psi^{**} = \psi$ (for a proof see Rockafellar, 1970, Corollary 12.2.1).

In this work we use several examples of closed proper convex functions. The first of them is relative entropy, viewed as a function of its first argument and extended to $\mathbb{R}^{\mathcal{X}}$ as follows:

$$\psi(p) = \begin{cases} \mathrm{D}(p \parallel q_0) & \text{if } p \in \Delta \\ \infty & \text{otherwise} \end{cases}$$

where $q_0 \in \Delta$ is assumed fixed. The conjugate of relative entropy is the log partition function

$$\psi^*(r) = \ln\left(\textstyle\sum_{x\in\mathcal{X}} q_0(x) e^{r(x)}\right)$$

where $r \in \mathbb{R}^{\mathcal{X}}$ and its components are denoted by $r(x)$.

The second example is the unnormalized relative entropy

$$\widetilde{\mathrm{D}}(p \parallel q_0) = \textstyle\sum_{x\in X} \left[p(x) \ln\left(\frac{p(x)}{q_0(x)}\right) - p(x) + q_0(x) \right] \ .$$

Fixing $q_0 \in [0,\infty)^X$, it can be extended to a closed proper convex function of its first argument:

$$\psi(p) = \begin{cases} \widetilde{\mathrm{D}}(p \parallel q_0) & \text{if } p(x) \geq 0 \text{ for all } x \in X \\ \infty & \text{otherwise.} \end{cases}$$

The conjugate of unnormalized relative entropy is a scaled exponential shifted to the origin:

$$\psi^*(r) = \textstyle\sum_{x\in X} q_0(x)(e^{r(x)} - 1) \ .$$

Both relative entropy and unnormalized relative entropy are examples of Bregman divergences (Bregman, 1967) which generalize some common distance measures including the squared Euclidean distance. We use two properties satisfied by any Bregman divergence $\mathrm{B}(\cdot \parallel \cdot)$:

(B1) $\mathrm{B}(\mathbf{a} \parallel \mathbf{b}) \geq 0$,
(B2) if $\mathrm{B}(\mathbf{a}_t \parallel \mathbf{b}_t) \to 0$ and $\mathbf{b}_t \to \mathbf{b}^\star$ then $\mathbf{a}_t \to \mathbf{b}^\star$.

It is not too difficult to check these properties explicitly both for relative entropy and unnormalized relative entropy.

Another example of a closed proper convex function is an *indicator function* of a closed convex set $C \subseteq \mathbb{R}^n$, denoted by I_C, which equals 0 when its argument lies in C and infinity otherwise. We will also use $\mathrm{I}(\mathbf{u} \in C)$ to denote $\mathrm{I}_C(\mathbf{u})$. The conjugate of an indicator function is a *support function*. For $C = \{\mathbf{u}_0\}$, we obtain $\mathrm{I}^*_{\{\mathbf{u}_0\}}(\boldsymbol{\lambda}) = \boldsymbol{\lambda} \cdot \mathbf{u}_0$. For a box $R = \{\mathbf{u} : |u_j| \leq \beta_j \text{ for all } j\}$, we obtain an ℓ_1-style conjugate $\mathrm{I}^*_R(\boldsymbol{\lambda}) = \sum_j \beta_j |\lambda_j|$. For a Euclidean ball $B = \{\mathbf{u} : \|\mathbf{u}\|_2 \leq \beta\}$, we obtain an ℓ_2-style conjugate, $\mathrm{I}^*_B(\boldsymbol{\lambda}) = \beta \|\boldsymbol{\lambda}\|_2$.

The final example is a square of the Euclidean norm $\psi(\mathbf{u}) = \|\mathbf{u}\|_2^2/(2\alpha)$, whose conjugate is also a square of the Euclidean norm $\psi^*(\boldsymbol{\lambda}) = \alpha\|\boldsymbol{\lambda}\|_2^2/2$.

The following identities can be proved from the definition of the conjugate function:

$$\text{if } \varphi(\mathbf{u}) = a\psi(b\mathbf{u} + \mathbf{c}) \qquad \text{then } \varphi^*(\boldsymbol{\lambda}) = a\psi^*(\boldsymbol{\lambda}/(ab)) - \boldsymbol{\lambda} \cdot \mathbf{c}/b \ , \tag{3}$$

$$\text{if } \varphi(\mathbf{u}) = \textstyle\sum_j \varphi_j(u_j) \qquad \text{then } \varphi^*(\boldsymbol{\lambda}) = \textstyle\sum_j \varphi_j^*(\lambda_j) \tag{4}$$

where $a > 0, b \neq 0$ and $\mathbf{c} \in \mathbb{R}^n$ are constants, and u_j, λ_j refer to the components of $\mathbf{u}, \boldsymbol{\lambda}$.

We conclude with a version of *Fenchel's Duality Theorem* which relates a convex minimization problem to a concave maximization problem using conjugates. The following result is essentially Corollary 31.2.1 of Rockafellar (1970) under a stronger set of assumptions.

Theorem 1 (Fenchel's Duality). *Let* $\psi : \mathbb{R}^n \to (-\infty, \infty]$ *and* $\varphi : \mathbb{R}^m \to (-\infty, \infty]$ *be closed proper convex functions and* $\mathbf{A}$ *a real-valued* $m \times n$ *matrix. Assume that* $\mathrm{dom}\,\psi^* = \mathbb{R}^n$ *or* $\mathrm{dom}\,\varphi = \mathbb{R}^m$. *Then*

$$\inf_{\mathbf{u}} \left[\psi(\mathbf{u}) + \varphi(\mathbf{A}\mathbf{u})\right] = \sup_{\boldsymbol{\lambda}} \left[-\psi^*(\mathbf{A}^\top \boldsymbol{\lambda}) - \varphi^*(-\boldsymbol{\lambda})\right] \ .$$

We refer to the minimization over $\mathbf{u}$ as the primal problem and the maximization over $\boldsymbol{\lambda}$ as the dual problem. When no ambiguity arises, we also refer to the minimization over $\boldsymbol{\lambda}$ of the negative dual objective as the dual problem. We call $\mathbf{u}$ a primal feasible point if the primal objective is finite at $\mathbf{u}$ and analogously define a dual feasible point.

4. Generalized Maximum Entropy

In this paper we study a generalized maxent problem

$$\mathcal{P}: \quad \min_{p\in\Delta}\bigl[\mathrm{D}(p \parallel q_0) + \mathrm{U}(p[\mathbf{f}])\bigr]$$

where $\mathrm{U} : \mathbb{R}^n \to (-\infty, \infty]$ is an arbitrary closed proper convex function. It is viewed as a *potential* for the maxent problem. We further assume that q_0 is positive on $\mathcal{X}$, that is, $\mathrm{D}(p \parallel q_0)$ is finite for all $p \in \Delta$ (otherwise we could restrict $\mathcal{X}$ to the support of q_0), and there exists a distribution whose vector of feature expectations is a feasible point of U (this is typically satisfied by the empirical distribution). These two conditions imply that the problem $\mathcal{P}$ is feasible.

The definition of generalized maxent captures many cases of interest including basic maxent, ℓ_1-regularized maxent and ℓ_2^2-regularized maxent. Basic maxent is obtained by using a point indicator potential $\mathrm{U}^{(0)}(\mathbf{u}) = \mathrm{I}(\mathbf{u} = \tilde{\pi}[\mathbf{f}])$. The ℓ_1-regularized version of maxent, as shown by Kazama and Tsujii (2003), corresponds to the relaxation of equality constraints to box constraints

$$|\tilde{\pi}[f_j] - p[f_j]| \le \beta_j \ .$$

This choice can be motivated by an observation that we do not expect $\tilde{\pi}[f_j]$ to be *equal* to $\pi[f_j]$ but only close to it. Box constraints are represented by the potential $\mathrm{U}^{(1)}(\mathbf{u}) = \mathrm{I}(|\tilde{\pi}[f_j] - u_j| \le \beta_j \text{ for all } j)$. Finally, as pointed out by Chen and Rosenfeld (2000) and Lebanon and Lafferty (2001), ℓ_2^2-regularized maxent is obtained using the potential $\mathrm{U}^{(2)}(\mathbf{u}) = \|\tilde{\pi}[\mathbf{f}] - \mathbf{u}\|_2^2/(2\alpha)$ which incurs an ℓ_2^2-style penalty for deviating from empirical averages.

The primal objective of generalized maxent will be referred to as P:

$$P(p) = \mathrm{D}(p \parallel q_0) + \mathrm{U}(p[\mathbf{f}]) \ .$$

Note that P attains its minimum over Δ, because Δ is compact and P is lower semi-continuous. The minimizer of P is unique by strict convexity of $\mathrm{D}(p \parallel q_0)$.

To derive the dual of $\mathcal{P}$, define the matrix $\mathbf{F}_{jx} = f_j(x)$ and use Fenchel's duality:

$$\begin{aligned}
\min_{p\in\Delta}[\mathrm{D}(p \parallel q_0) + \mathrm{U}(p[\mathbf{f}])] &= \min_{p\in\Delta}[\mathrm{D}(p \parallel q_0) + \mathrm{U}(\mathbf{F}p)] \\
&= \sup_{\boldsymbol{\lambda}\in\mathbb{R}^n}\Bigl[-\ln\Bigl(\textstyle\sum_{x\in\mathcal{X}} q_0(x)\exp\bigl\{(\mathbf{F}^\top\boldsymbol{\lambda})_x\bigr\}\Bigr) - \mathrm{U}^*(-\boldsymbol{\lambda})\Bigr] && (5) \\
&= \sup_{\boldsymbol{\lambda}\in\mathbb{R}^n}[-\ln Z_{\boldsymbol{\lambda}} - \mathrm{U}^*(-\boldsymbol{\lambda})] \ . && (6)
\end{aligned}$$

In Equation (5), we apply Theorem 1. We use $(\mathbf{F}^\top\boldsymbol{\lambda})_x$ to denote the entry of $\mathbf{F}^\top\boldsymbol{\lambda}$ indexed by x. In Equation (6), we note that $(\mathbf{F}^\top\boldsymbol{\lambda})_x = \boldsymbol{\lambda}\cdot\mathbf{f}(x)$ and thus the expression inside the logarithm is the normalization constant of $q_{\boldsymbol{\lambda}}$. The dual objective will be referred to as Q:

$$Q(\boldsymbol{\lambda}) = -\ln Z_{\boldsymbol{\lambda}} - \mathrm{U}^*(-\boldsymbol{\lambda}) \ .$$

There are two formal differences between generalized maxent and basic maxent. The first difference is that the constraints of the basic primal (1) are stated relative to the empirical expectations whereas the potential of the generalized primal $\mathcal{P}$ makes no reference to $\tilde{\pi}[\mathbf{f}]$. This difference is only superficial. It is possible to "hard-wire" the distribution $\tilde{\pi}$ in the potential U, as we saw on

	potential (absolute and relative)	conjugate potential
generalized maxent:		
$\mathrm{U}(\mathbf{u})$	$\mathrm{U}(\mathbf{u})$	$\mathrm{U}^*(\boldsymbol{\lambda})$
$\mathrm{U}_r(\mathbf{u})$	$\mathrm{U}(r[\mathbf{f}]-\mathbf{u})$	$\mathrm{U}^*(-\boldsymbol{\lambda})+\boldsymbol{\lambda}\cdot r[\mathbf{f}]$
$\mathrm{U}_{\tilde{\pi}}(\mathbf{u})$	$\mathrm{U}(\tilde{\pi}[\mathbf{f}]-\mathbf{u})$	$\mathrm{U}^*(-\boldsymbol{\lambda})+\boldsymbol{\lambda}\cdot \tilde{\pi}[\mathbf{f}]$
basic constraints:		
$\mathrm{U}^{(0)}(\mathbf{u})$	$\mathrm{I}(\mathbf{u}=\tilde{\pi}[\mathbf{f}])$	$\boldsymbol{\lambda}\cdot\tilde{\pi}[\mathbf{f}]$
$\mathrm{U}_r^{(0)}(\mathbf{u})$	$\mathrm{I}(\mathbf{u}=r[\mathbf{f}]-\tilde{\pi}[\mathbf{f}])$	$\boldsymbol{\lambda}\cdot(r[\mathbf{f}]-\tilde{\pi}[\mathbf{f}])$
$\mathrm{U}_{\tilde{\pi}}^{(0)}(\mathbf{u})$	$\mathrm{I}(\mathbf{u}=\mathbf{0})$	0
box constraints:		
$\mathrm{U}^{(1)}(\mathbf{u})$	$\mathrm{I}(\lvert\tilde{\pi}[f_j]-u_j\rvert\le\beta_j \text{ for all } j)$	$\boldsymbol{\lambda}\cdot\tilde{\pi}[\mathbf{f}]+\sum_j\beta_j\lvert\lambda_j\rvert$
$\mathrm{U}_r^{(1)}(\mathbf{u})$	$\mathrm{I}(\lvert u_j-(r[f_j]-\tilde{\pi}[f_j])\rvert\le\beta_j \text{ for all } j)$	$\boldsymbol{\lambda}\cdot(r[\mathbf{f}]-\tilde{\pi}[\mathbf{f}])+\sum_j\beta_j\lvert\lambda_j\rvert$
$\mathrm{U}_{\tilde{\pi}}^{(1)}(\mathbf{u})$	$\mathrm{I}(\lvert u_j\rvert\le\beta_j \text{ for all } j)$	$\sum_j\beta_j\lvert\lambda_j\rvert$
ℓ_2^2 penalty:		
$\mathrm{U}^{(2)}(\mathbf{u})$	$\lVert\tilde{\pi}[\mathbf{f}]-\mathbf{u}\rVert_2^2/(2\alpha)$	$\boldsymbol{\lambda}\cdot\tilde{\pi}[\mathbf{f}]+\alpha\lVert\boldsymbol{\lambda}\rVert_2^2/2$
$\mathrm{U}_r^{(2)}(\mathbf{u})$	$\lVert\mathbf{u}-(r[\mathbf{f}]-\tilde{\pi}[\mathbf{f}])\rVert_2^2/(2\alpha)$	$\boldsymbol{\lambda}\cdot(r[\mathbf{f}]-\tilde{\pi}[\mathbf{f}])+\alpha\lVert\boldsymbol{\lambda}\rVert_2^2/2$
$\mathrm{U}_{\tilde{\pi}}^{(2)}(\mathbf{u})$	$\lVert\mathbf{u}\rVert_2^2/(2\alpha)$	$\alpha\lVert\boldsymbol{\lambda}\rVert_2^2/2$

Table 1: Absolute and relative potentials, and their conjugates for various versions of maxent.

the example of $\mathrm{U}^{(0)}$. In the latter case, it would be more correct, but perhaps overly pedantic and somewhat clumsy, to make the dependence of the potential on $\tilde{\pi}$ explicit and use the notation $\mathrm{U}^{(0),\tilde{\pi}}$.

The second difference, which seems more significant, is the difference between the duals. The objective of the basic dual (2) equals the log loss relative to the empirical distribution $\tilde{\pi}$, but the log loss does not appear in the generalized dual. However, we will see that the generalized dual can be expressed in terms of the log loss. In fact, it can be expressed in terms of the log loss relative to an arbitrary distribution, including the empirical distribution $\tilde{\pi}$ as well as the unknown distribution π.

We next describe *shifting*, the transformation of an "absolute" potential to a "relative" potential. Shifting is a technical tool which will simplify some of the proofs in Sections 5 and 6, and will also be used to rewrite the generalized dual in terms of the log loss.

4.1 Shifting

For an arbitrary distribution r and a potential U, let U_r denote the function

$$\mathrm{U}_r(\mathbf{u})=\mathrm{U}(r[\mathbf{f}]-\mathbf{u}) \ .$$

This function will be referred to as the *potential relative to r* or simply the *relative potential*. The original potential U will be in contrast referred to as the *absolute potential*. In Table 1, we list potentials discussed so far, alongside their versions relative to an arbitrary distribution r, and relative to $\tilde{\pi}$ in particular.

From the definition of a relative potential, we see that the absolute potential can be expressed as $\mathrm{U}(\mathbf{u})=\mathrm{U}_r(r[\mathbf{f}]-\mathbf{u})$. Thus, it is possible to implicitly define a potential U by defining a relative potential U_r for a particular distribution r. The potentials $\mathrm{U}^{(0)}, \mathrm{U}^{(1)}, \mathrm{U}^{(2)}$ of basic maxent, maxent with box constraints, and maxent with ℓ_2^2 penalty could thus have been specified by defining $\mathrm{U}_{\tilde{\pi}}^{(0)}(\mathbf{u})=\mathrm{I}(\mathbf{u}=\mathbf{0})$, $\mathrm{U}_{\tilde{\pi}}^{(1)}(\mathbf{u})=\mathrm{I}(\lvert u_j\rvert\le\beta_j \text{ for all } j)$ and $\mathrm{U}_{\tilde{\pi}}^{(2)}(\mathbf{u})=\lVert\mathbf{u}\rVert_2^2/(2\alpha)$.

The conjugate of a relative potential, the *conjugate relative potential*, is obtained, according to Equation (3), by adding a linear function to the conjugate of U:

$$\mathrm{U}_r^*(\boldsymbol{\lambda}) = \mathrm{U}^*(-\boldsymbol{\lambda}) + \boldsymbol{\lambda}\cdot r[\mathbf{f}] \ . \tag{7}$$

Table 1 lists $\mathrm{U}^{(0)*}$, $\mathrm{U}^{(1)*}$, $\mathrm{U}^{(2)*}$, and the conjugates of the corresponding relative potentials.

4.2 The Generalized Dual as the Minimization of a Regularized Log Loss

We will now show how the dual objective $Q(\boldsymbol{\lambda})$ can be expressed in terms of the log loss relative to an arbitrary distribution r. This will highlight how the dual of the generalized maxent extends the dual of the basic maxent. Using Equation (7), we rewrite $Q(\boldsymbol{\lambda})$ as follows:

$$\begin{aligned} Q(\boldsymbol{\lambda}) &= -\ln Z_{\boldsymbol{\lambda}} - \mathrm{U}^*(-\boldsymbol{\lambda}) = -\ln Z_{\boldsymbol{\lambda}} - \mathrm{U}_r^*(\boldsymbol{\lambda}) + \boldsymbol{\lambda}\cdot r[\mathbf{f}] \\ &= -r[\ln q_0] + r[\ln q_0 + \boldsymbol{\lambda}\cdot\mathbf{f} - \ln Z_{\boldsymbol{\lambda}}] - \mathrm{U}_r^*(\boldsymbol{\lambda}) \\ &= \mathrm{L}_r(\mathbf{0}) - \mathrm{L}_r(\boldsymbol{\lambda}) - \mathrm{U}_r^*(\boldsymbol{\lambda}) \ . \end{aligned} \tag{8}$$

Since the first term in Equation (8) is a constant independent of $\boldsymbol{\lambda}$, the maximization of $Q(\boldsymbol{\lambda})$ is equivalent to the minimization of $\mathrm{L}_r(\boldsymbol{\lambda}) + \mathrm{U}_r^*(\boldsymbol{\lambda})$. Setting $r = \tilde{\pi}$ we obtain a dual analogous to the basic dual (2):

$$\mathcal{Q}_{\tilde{\pi}}: \quad \inf_{\boldsymbol{\lambda}\in\mathbb{R}^n} \left[\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \mathrm{U}_{\tilde{\pi}}^*(\boldsymbol{\lambda})\right] \ .$$

From Equation (8), it follows that the $\boldsymbol{\lambda}$ minimizing $\mathrm{L}_r(\boldsymbol{\lambda}) + \mathrm{U}_r^*(\boldsymbol{\lambda})$ does not depend on a particular choice of r. As a result, the minimizer of $\mathcal{Q}_{\tilde{\pi}}$ is also the minimizer of $\mathrm{L}_\pi(\boldsymbol{\lambda}) + \mathrm{U}_\pi^*(\boldsymbol{\lambda})$. This observation will be used in Section 5 to prove performance guarantees.

The objective of $\mathcal{Q}_{\tilde{\pi}}$ has two terms. The first of them is the empirical log loss. The second one is the regularization term penalizing "complex" solutions. The regularization term need not be non-negative and it does not necessarily increase with any norm of $\boldsymbol{\lambda}$. On the other hand, it is a proper closed convex function and if $\tilde{\pi}$ is feasible then by Fenchel's inequality the regularization is bounded from below by $-\mathrm{U}_{\tilde{\pi}}(\mathbf{0})$. From a Bayesian perspective, $\mathrm{U}_{\tilde{\pi}}^*$ corresponds to negative log of the prior, and minimizing $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \mathrm{U}_{\tilde{\pi}}^*(\boldsymbol{\lambda})$ is equivalent to maximizing the posterior.

In the case of basic maxent, we obtain $\mathrm{U}_{\tilde{\pi}}^{(0)*}(\boldsymbol{\lambda}) = 0$ and recover the basic dual. For the box potential, we obtain $\mathrm{U}_{\tilde{\pi}}^{(1)*}(\boldsymbol{\lambda}) = \sum_j \beta_j |\lambda_j|$, which corresponds to an ℓ_1-style regularization and a Laplace prior. For the ℓ_2^2 potential, we obtain $\mathrm{U}_{\tilde{\pi}}^{(2)*}(\boldsymbol{\lambda}) = \alpha\|\boldsymbol{\lambda}\|_2^2/2$, which corresponds to an ℓ_2^2-style regularization and a Gaussian prior.

In all the cases discussed in this paper, it is natural to consider the dual objective relative to $\tilde{\pi}$ as we have seen in the previous examples. In other cases, the empirical distribution $\tilde{\pi}$ need not be available, and there may be no natural distribution relative to which a potential could be specified, yet it is possible to define a meaningful absolute potential (Dudík et al., 2005; Dudík and Schapire, 2006). To capture the more general case, we formulate the generalized maxent using the absolute potential.

4.3 Maxent Duality

We know from Equation (6) that the generalized maxent primal and dual have equal *values*. In this section, we show the equivalence of the primal and dual *optimizers*. Specifically, we show that the maxent primal $\mathcal{P}$ is solved by the Gibbs distribution whose parameter vector $\boldsymbol{\lambda}$ solves the dual (possibly in a limit). This parallels the result of Della Pietra, Della Pietra, and Lafferty (1997) for the basic maxent and gives additional motivation for the view of the dual objective as the regularized log loss.

Theorem 2 (Maxent Duality). *Let q_0, U, P, Q be as above. Then*

$$\min_{p\in\Delta} P(p) = \sup_{\boldsymbol{\lambda}\in\mathbb{R}^n} Q(\boldsymbol{\lambda}) \ . \tag{9}$$

Moreover, for a sequence $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \ldots$ such that

$$\lim_{t\to\infty} Q(\boldsymbol{\lambda}_t) = \sup_{\boldsymbol{\lambda}\in\mathbb{R}^n} Q(\boldsymbol{\lambda})$$

the sequence of $q_t = q_{\boldsymbol{\lambda}_t}$ has a limit and

$$P\left(\lim_{t\to\infty} q_t\right) = \min_{p\in\Delta} P(p) \ . \tag{10}$$

Proof. Equation (9) is a consequence of Fenchel's duality as was shown earlier. It remains to prove Equation (10). We will use an alternative expression for the dual objective. Let r be an arbitrary distribution. Adding and subtracting $\mathrm{H}(r)$ from Equation (8) yields

$$Q(\boldsymbol{\lambda}) = -\mathrm{D}(r \parallel q_{\boldsymbol{\lambda}}) + \mathrm{D}(r \parallel q_0) - \mathrm{U}_r^*(\boldsymbol{\lambda}) \ . \tag{11}$$

Let $\hat{p}$ be the minimizer of P and $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \ldots$ maximize Q in the limit. Then

$$\begin{aligned}\mathrm{D}(\hat{p} \parallel q_0) + \mathrm{U}_{\hat{p}}(\mathbf{0}) = P(\hat{p}) &= \sup_{\boldsymbol{\lambda}\in\mathbb{R}^n} Q(\boldsymbol{\lambda}) = \lim_{t\to\infty} Q(\boldsymbol{\lambda}_t) \\ &= \lim_{t\to\infty}\left[-\mathrm{D}(\hat{p} \parallel q_t) + \mathrm{D}(\hat{p} \parallel q_0) - \mathrm{U}_{\hat{p}}^*(\boldsymbol{\lambda}_t)\right] \ .\end{aligned}$$

Denoting terms with the limit 0 by $o(1)$ and rearranging yields

$$\mathrm{U}_{\hat{p}}(\mathbf{0}) + \mathrm{U}_{\hat{p}}^*(\boldsymbol{\lambda}_t) = -\mathrm{D}(\hat{p} \parallel q_t) + o(1) \ .$$

The left-hand side is non-negative by Fenchel's inequality, so $\mathrm{D}(\hat{p} \parallel q_t) \to 0$ by the non-negativity of relative entropy. Therefore, by property (B2), every convergent subsequence of $q_1, q_2, \ldots$ has the limit $\hat{p}$. Since the q_t's come from the compact set Δ, we obtain $q_t \to \hat{p}$. ■

Thus, in order to solve the primal, it suffices to find a sequence of $\boldsymbol{\lambda}$'s maximizing the dual. This will be the goal of algorithms in Sections 6 and 7.

5. Bounding the Loss on the Target Distribution

In this section, we derive bounds on the performance of generalized maxent relative to the true distribution π. That is, we are able to bound $\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}})$ in terms of $\mathrm{L}_\pi(\boldsymbol{\lambda}^\star)$ when $q_{\hat{\boldsymbol{\lambda}}}$ maximizes the dual objective Q and $q_{\boldsymbol{\lambda}^\star}$ is either an arbitrary Gibbs distribution, or in some cases, a Gibbs distribution with a bounded norm of $\boldsymbol{\lambda}^\star$. In particular, bounds hold for the Gibbs distribution minimizing the true loss (in some cases, among Gibbs distributions with a bounded norm of $\boldsymbol{\lambda}^\star$). Note that $\mathrm{D}(\pi \parallel q_{\boldsymbol{\lambda}})$ differs from $\mathrm{L}_\pi(\boldsymbol{\lambda})$ only by the constant term $\mathrm{H}(\pi)$, so identical bounds also hold for $\mathrm{D}(\pi \parallel q_{\hat{\boldsymbol{\lambda}}})$ in terms of $\mathrm{D}(\pi \parallel q_{\boldsymbol{\lambda}^\star})$.

Our results are stated for the case when the supremum of Q is attained at $\hat{\boldsymbol{\lambda}} \in \mathbb{R}^n$, but they easily extend to the case when the supremum is only attained in a limit. The crux of our method is the lemma below. Even though its proof is remarkably simple, it is sufficiently general to cover all the cases of interest.

Lemma 3. *Let $\hat{\boldsymbol{\lambda}}$ maximize Q. Then for an arbitrary Gibbs distribution $q_{\boldsymbol{\lambda}^\star}$*

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + 2\mathrm{U}(\pi[\mathbf{f}]) + \mathrm{U}^*(\boldsymbol{\lambda}^\star) + \mathrm{U}^*(-\boldsymbol{\lambda}^\star) \ , \tag{12}$$

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + 2\mathrm{U}_{\tilde{\pi}}(\tilde{\pi}[\mathbf{f}] - \pi[\mathbf{f}]) + \mathrm{U}^*_{\tilde{\pi}}(\boldsymbol{\lambda}^\star) + \mathrm{U}^*_{\tilde{\pi}}(-\boldsymbol{\lambda}^\star) \ , \tag{13}$$

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + (\boldsymbol{\lambda}^\star - \hat{\boldsymbol{\lambda}}) \cdot (\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]) + \mathrm{U}^*_{\tilde{\pi}}(\boldsymbol{\lambda}^\star) - \mathrm{U}^*_{\tilde{\pi}}(\hat{\boldsymbol{\lambda}}) \ . \tag{14}$$

Proof. Optimality of $\hat{\boldsymbol{\lambda}}$ with respect to $\mathrm{L}_\pi(\boldsymbol{\lambda}) + \mathrm{U}^*_\pi(\boldsymbol{\lambda}) = -Q(\boldsymbol{\lambda}) + const.$ yields

$$\begin{aligned} \mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) &\le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \mathrm{U}^*_\pi(\boldsymbol{\lambda}^\star) - \mathrm{U}^*_\pi(\hat{\boldsymbol{\lambda}}) \\ &\le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + (\boldsymbol{\lambda}^\star - \hat{\boldsymbol{\lambda}}) \cdot \pi[\mathbf{f}] + \mathrm{U}^*(-\boldsymbol{\lambda}^\star) - \mathrm{U}^*(-\hat{\boldsymbol{\lambda}}) \ . \end{aligned} \tag{15}$$

In Equation (15), we express U^*_π in terms of U^* using Equation (7). Now Equation (12) is obtained by applying Fenchel's inequality to the second term of Equation (15):

$$(\boldsymbol{\lambda}^\star - \hat{\boldsymbol{\lambda}}) \cdot \pi[\mathbf{f}] \le \mathrm{U}^*(\boldsymbol{\lambda}^\star) + \mathrm{U}(\pi[\mathbf{f}]) + \mathrm{U}^*(-\hat{\boldsymbol{\lambda}}) + \mathrm{U}(\pi[\mathbf{f}]) \ .$$

Equations (13) and (14) follow from Equations (12) and (15) by shifting potentials and their conjugates to $\tilde{\pi}$. ■

Remark. Notice that π and $\tilde{\pi}$ in the statement and the proof of the lemma can be replaced by arbitrary distributions p_1 and p_2.

A special case which we discuss in more detail is when U is an indicator of a closed convex set C, such as $\mathrm{U}^{(0)}$ and $\mathrm{U}^{(1)}$ of the previous section. In that case, the right hand side of Lemma 3.12 will be infinite unless $\pi[\mathbf{f}]$ lies in C. In order to apply Lemma 3.12, we ensure that $\pi[\mathbf{f}] \in C$ with high probability. Therefore, we choose C as a confidence region for $\pi[\mathbf{f}]$. If $\pi[\mathbf{f}] \in C$ then for any Gibbs distribution $q_{\boldsymbol{\lambda}^\star}$

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \mathrm{I}^*_C(\boldsymbol{\lambda}^\star) + \mathrm{I}^*_C(-\boldsymbol{\lambda}^\star) \ . \tag{16}$$

For a fixed $\boldsymbol{\lambda}^\star$ and a non-empty C, $\mathrm{I}^*_C(\boldsymbol{\lambda}^\star) + \mathrm{I}^*_C(-\boldsymbol{\lambda}^\star)$ is always non-negative and proportional to the size of C's projection onto a line in the direction $\boldsymbol{\lambda}^\star$. Thus, smaller confidence regions yield better performance guarantees.

A common method of obtaining confidence regions is to bound the difference between empirical averages and true expectations. There exists a huge array of techniques to achieve this. Before moving to specific examples, we state a general result which follows directly from Lemma 3.13 analogously to Equation (16).

Theorem 4. *Assume that $\tilde{\pi}[\mathbf{f}] - \pi[\mathbf{f}] \in C_0$ where C_0 is a closed convex set symmetric around the origin. Let $\hat{\boldsymbol{\lambda}}$ minimize $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \mathrm{I}^*_{C_0}(\boldsymbol{\lambda})$. Then for an arbitrary Gibbs distribution $q_{\boldsymbol{\lambda}^\star}$*

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + 2\mathrm{I}^*_{C_0}(\boldsymbol{\lambda}^\star) \ .$$

Proof. Setting $\mathrm{U}_{\tilde{\pi}}(\mathbf{u}) = \mathrm{I}_{C_0}(\mathbf{u})$ and assuming $\tilde{\pi}[\mathbf{f}] - \pi[\mathbf{f}] \in C_0$, we obtain by Lemma 3.13

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \mathrm{I}^*_{C_0}(\boldsymbol{\lambda}^\star) + \mathrm{I}^*_{C_0}(-\boldsymbol{\lambda}^\star) \ .$$

The result now follows by the symmetry of C_0, which implies the symmetry of I_{C_0}, which in turn implies the symmetry of $\mathrm{I}^*_{C_0}$. ■

5.1 Maxent with ℓ_1 Regularization

We now apply the foregoing general results to some specific cases of interest. To begin, we consider the box indicator $\mathrm{U}^{(1)}$ of Section 4. In this case it suffices to bound $|\tilde{\pi}[f_j] - \pi[f_j]|$ and use Theorem 4 to obtain a bound on the true loss $\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}})$. For instance, when the features are bounded, we can prove the following:

Corollary 5. *Assume that features $f_1, \dots, f_n$ are bounded in $[0,1]$. Let $\delta > 0$ and let $\hat{\boldsymbol{\lambda}}$ minimize $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \beta \|\boldsymbol{\lambda}\|_1$ with $\beta = \sqrt{\ln(2n/\delta)/(2m)}$. Then with probability at least $1-\delta$, for every Gibbs distribution $q_{\boldsymbol{\lambda}^\star}$,*

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \frac{\|\boldsymbol{\lambda}^\star\|_1}{\sqrt{m}} \sqrt{2\ln(2n/\delta)} \ .$$

Proof. By Hoeffding's inequality, for a fixed j, the probability that $|\tilde{\pi}[f_j] - \pi[f_j]|$ exceeds β is at most $2e^{-2\beta^2 m} = \delta/n$. By the union bound, the probability of this happening for any j is at most δ. The claim now follows immediately from Theorem 4. ■

Similarly, when the f_j's are selected from a possibly larger class of binary features with VC-dimension d, we can prove the following corollary. This will be the case, for instance, when using threshold features on k variables, a class with VC-dimension $O(\ln k)$.

Corollary 6. *Assume that features are binary with VC-dimension d. Let $\delta > 0$ and let $\hat{\boldsymbol{\lambda}}$ minimize $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \beta \|\boldsymbol{\lambda}\|_1$ with*

$$\beta = \sqrt{\frac{d\ln(em^2/d) + \ln(1/\delta) + \ln(4e^8)}{2m}} \ .$$

Then with probability at least $1-\delta$, for every Gibbs distribution $q_{\boldsymbol{\lambda}^\star}$,

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \frac{\|\boldsymbol{\lambda}^\star\|_1}{\sqrt{m}} \sqrt{2[d\ln(em^2/d) + \ln(1/\delta) + \ln(4e^8)]} \ .$$

Proof. Here, a uniform-convergence result of Devroye (1982), combined with Sauer's Lemma, can be used to argue that $|\tilde{\pi}[f_j] - \pi[f_j]| \le \beta$ for all f_j simultaneously with probability at least $1-\delta$. ■

The final result for ℓ_1-regularized maxent is motivated by the Central Limit Theorem approximation $|\tilde{\pi}[f_j] - \pi[f_j]| = O(\sigma[f_j]/\sqrt{m})$, where $\sigma[f_j]$ is the standard deviation of f_j under π. We bound $\sigma[f_j]$ from above using McDiarmid's inequality for the empirical estimate of variance

$$\tilde{\sigma}^2[f_j] = \frac{m\big(\tilde{\pi}[f_j^2] - \tilde{\pi}[f_j]^2\big)}{m-1} \ ,$$

and then obtain non-asymptotic bounds on $|\tilde{\pi}[f_j] - \pi[f_j]|$ by Bernstein's inequality (for a complete proof see Appendix A).

We believe that this type of result may in practice be more useful than Corollaries 5 and 6, because it allows differentiation between features depending on empirical error estimates computed from the sample data. Motivated by Corollary 7 below, in Section 8 we describe experiments that use $\beta_j = \beta_0 \tilde{\sigma}[f_j]/\sqrt{m}$, where β_0 is a single tuning constant. This approach is equivalent to using features scaled to the unit sample variance, that is, features $f'_j(x) = f_j(x)/\tilde{\sigma}[f_j]$, and a regularization parameter independent of features, $\beta'_j = \beta_0/\sqrt{m}$, as is a common practice in statistics. Corollary 7 justifies this practice and also suggests replacing the sample variance by a slightly larger value $\tilde{\sigma}^2[f_j] + O(1/\sqrt{m})$.

Corollary 7. *Assume that features $f_1, \dots, f_n$ are bounded in $[0,1]$. Let $\delta > 0$ and let $\hat{\boldsymbol{\lambda}}$ minimize $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \sum_j \beta_j |\lambda_j|$ with*

$$\beta_j = \sqrt{\frac{2\ln(4n/\delta)}{m}} \cdot \sqrt{\tilde{\sigma}^2[f_j] + \sqrt{\frac{\ln(2n/\delta)}{2m}} + \frac{\ln(4n/\delta)}{18m}} + \frac{\ln(4n/\delta)}{3m} \ .$$

Then with probability at least $1-\delta$, for every Gibbs distribution $q_{\boldsymbol{\lambda}^\star}$,

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + 2\sum_j \beta_j |\lambda_j^\star| \ .$$

Corollaries of this section show that the difference in performance between the distribution computed by minimizing ℓ_1-regularized log loss and the best Gibbs distribution becomes small rapidly as the number of samples m increases. Note that this difference depends only moderately on the number or complexity of features.

Another feature of ℓ_1 regularization is that it induces sparsity (Tibshirani, 1996). Note that a maxent solution $\hat{\boldsymbol{\lambda}}$ is "truly" sparse, that is, some of its components are "truly" zero, only if they remain zero under perturbations in the regularization parameters β_j and the expectations $\tilde{\pi}[f_j]$; in other words, the fact that the components of $\hat{\boldsymbol{\lambda}}$ are zero is not just a lucky coincidence. To see how ℓ_1 regularization induces this property, notice that its partial derivatives are discontinuous at $\lambda_j = 0$. As a consequence, if the regularized log loss is uniquely minimized at a point where the j_0-th component $\hat{\lambda}_{j_0}$ equals zero, then the optimal $\hat{\lambda}_{j_0}$ will remain zero even if the parameters β_j and the expectations $\tilde{\pi}[f_j]$ are slightly perturbed.

5.2 Maxent with Smoothed ℓ_1 Regularization

While the guarantees for ℓ_1-style regularization have many favorable properties, the fact that the ℓ_1 norm is not strictly convex and its first derivative is discontinuous at zero may sometimes be problematic. The lack of strict convexity may lead to infinitely many $\boldsymbol{\lambda}$'s optimizing the dual objective,[2] and the discontinuous derivatives may cause problems in certain convex optimization algorithms. To prevent these problems, smooth approximations of ℓ_1 regularization may be necessary.

In this section, we analyze a smooth approximation similar to one used by Dekel, Shalev-Shwartz, and Singer (2003):

$$\mathrm{U}_{\tilde{\pi}}^{(\approx 1)*}(\boldsymbol{\lambda}) = \sum_j \alpha_j \beta_j \ln\cosh(\lambda_j/\alpha_j) = \sum_j \alpha_j \beta_j \ln\left(\frac{e^{\lambda_j/\alpha_j} + e^{-\lambda_j/\alpha_j}}{2}\right) \ .$$

Constants $\alpha_j > 0$ control the tightness of fit to the ℓ_1 norm while constants $\beta_j \ge 0$ control scaling. Note that $\cosh x \le e^{|x|}$ hence

$$\mathrm{U}_{\tilde{\pi}}^{(\approx 1)*}(\boldsymbol{\lambda}) \le \sum_j \alpha_j \beta_j \ln e^{|\lambda_j|/\alpha_j} = \sum_j \alpha_j \beta_j |\lambda_j|/\alpha_j = \sum_j \beta_j |\lambda_j| \ . \tag{17}$$

The potential corresponding to $\mathrm{U}_{\tilde{\pi}}^{(\approx 1)*}$ is

$$\mathrm{U}_{\tilde{\pi}}^{(\approx 1)}(\mathbf{u}) = \sum_j \alpha_j \beta_j \mathrm{D}\left(\frac{1 + u_j/\beta_j}{2} \,\middle\|\, \frac{1}{2}\right)$$

2. This may only happen if features are not linearly independent.

where $\mathrm{D}(a \parallel b)$ is a shorthand for $\mathrm{D}((a,1-a) \parallel (b,1-b))$ (for a derivation of $\mathrm{U}_{\tilde{\pi}}^{(\approx 1)}$ see Appendix B). This potential can be viewed as a smooth upper bound on the box potential $\mathrm{U}_{\tilde{\pi}}^{(1)}$ in the sense that the gradient of $\mathrm{U}_{\tilde{\pi}}^{(\approx 1)}$ is continuous on the interior of the effective domain of $\mathrm{U}_{\tilde{\pi}}^{(1)}$ and its norm approaches ∞ on the border. Note that if $|u_j| \le \beta_j$ for all j then $\mathrm{D}\big(\frac{1+u_j/\beta_j}{2} \parallel \frac{1}{2}\big) \le \mathrm{D}\big(0 \parallel \frac{1}{2}\big) = \ln 2$ and hence

$$\mathrm{U}_{\tilde{\pi}}^{(\approx 1)}(\mathbf{u}) \le (\ln 2)\textstyle\sum_j \alpha_j \beta_j \ . \tag{18}$$

Applying bounds (17) and (18) in Lemma 3.13 we obtain an analog of Theorem 4.

Theorem 8. *Assume that for each j, $|\tilde{\pi}[f_j] - \pi[f_j]| \le \beta_j$. Let $\hat{\boldsymbol{\lambda}}$ minimize $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \mathrm{U}_{\tilde{\pi}}^{(\approx 1)*}(\boldsymbol{\lambda})$. Then for an arbitrary Gibbs distribution $q_{\boldsymbol{\lambda}^\star}$*

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + 2\textstyle\sum_j \beta_j |\lambda_j^\star| + (2\ln 2)\sum_j \alpha_j \beta_j \ .$$

To obtain guarantees analogous to those of ℓ_1-regularized maxent, it suffices to choose sufficiently small α_j. For example, in order to perform well relative to distributions $q_{\boldsymbol{\lambda}^\star}$ with $\sum_j \beta_j |\lambda_j^\star| \le L$, it suffices to choose $\alpha_j = (\varepsilon L)/(n\beta_j \ln 2)$ and obtain

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + 2(1+\varepsilon)L \ .$$

For example, we can derive an analog of Corollary 5. We relax the constraint that features are bounded in $[0,1]$ and, instead, provide a guarantee in terms of the ℓ_∞ diameter of the feature space.

Corollary 9. *Let $D_\infty = \sup_{x,x' \in \mathcal{X}} \|\mathbf{f}(x) - \mathbf{f}(x')\|_\infty$ be the ℓ_∞ diameter of $\mathbf{f}(\mathcal{X})$. Let $\delta, \varepsilon, L_1 > 0$ and let $\hat{\boldsymbol{\lambda}}$ minimize $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \alpha\beta \sum_j \ln\cosh(\lambda_j/\alpha)$ with*

$$\alpha = \frac{\varepsilon L_1}{n \ln 2} \ , \qquad \beta = D_\infty \sqrt{\frac{\ln(2n/\delta)}{2m}} \ .$$

Then with probability at least $1-\delta$

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \inf_{\|\boldsymbol{\lambda}^\star\|_1 \le L_1} \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \frac{(1+\varepsilon)L_1 D_\infty}{\sqrt{m}} \cdot \sqrt{2\ln(2n/\delta)} \ .$$

Thus, maxent with smoothed ℓ_1 regularization performs almost as well as ℓ_1-regularized maxent, provided that we specify an upper bound on the ℓ_1 norm of $\boldsymbol{\lambda}^\star$ in advance. As a result of removing discontinuities in the gradient, smoothed ℓ_1 regularization lacks the sparsity inducing properties of ℓ_1 regularization.

As $\alpha \to 0$, the guarantees for smoothed ℓ_1 regularization converge to those for ℓ_1 regularization, but at the price of reducing smoothness of the objective in some regions and increasing its flatness in others. For many methods of convex optimization, this leads to a worse runtime. For example, the number of iterations of gradient descent increases with an increasing condition number of the Hessian of the objective, which in our case grows as $\alpha \to 0$. Similarly, the number of iterations of Newton's method depends both on the condition number and the Lipschitz constant of the Hessian, both of which increase as $\alpha \to 0$. Thus, in choosing α, we trade an improvement in performance guarantees for an increase in runtime.

5.3 Maxent with ℓ_2 Regularization

In some cases, tighter performance guarantees are obtained by using confidence regions which take the shape of a Euclidean ball. More specifically, we consider the potential and conjugate

$$\mathrm{U}_{\tilde{\pi}}^{(\sqrt{2})}(\mathbf{u}) = \begin{cases} 0 & \text{if } \|\mathbf{u}\|_2 \le \beta \\ \infty & \text{otherwise} \end{cases} \quad , \qquad \mathrm{U}_{\tilde{\pi}}^{(\sqrt{2})*}(\boldsymbol{\lambda}) = \beta\|\boldsymbol{\lambda}\|_2 \ .$$

We first derive an ℓ_2 version of Hoeffding's inequality (Lemma 10 below, proved in Appendix C) and then use Theorem 4 to obtain performance guarantees.

Lemma 10. *Let $D_2 = \sup_{x,x' \in \mathcal{X}} \|\mathbf{f}(x) - \mathbf{f}(x')\|_2$ be the ℓ_2 diameter of $\mathbf{f}(\mathcal{X})$ and let $\delta > 0$. Then with probability at least $1 - \delta$*

$$\|\tilde{\pi}[\mathbf{f}] - \pi[\mathbf{f}]\|_2 \le \frac{D_2}{\sqrt{2m}}\left[1 + \sqrt{\ln(1/\delta)}\right] \ .$$

Theorem 11. *Let D_2 be the ℓ_2 diameter of $\mathbf{f}(\mathcal{X})$. Let $\delta > 0$ and let $\hat{\boldsymbol{\lambda}}$ minimize $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \beta\|\boldsymbol{\lambda}\|_2$ with $\beta = D_2\left[1 + \sqrt{\ln(1/\delta)}\right]/\sqrt{2m}$. Then with probability at least $1 - \delta$, for every Gibbs distribution $q_{\boldsymbol{\lambda}^\star}$,*

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \frac{\|\boldsymbol{\lambda}^\star\|_2 D_2}{\sqrt{m}}\left(\sqrt{2} + \sqrt{2\ln(1/\delta)}\right) \ .$$

Unlike results of the previous sections, this bound does not explicitly depend on the number of features and only grows with the ℓ_2 diameter of the feature space. The ℓ_2 diameter is small for example when the feature space consists of sparse binary vectors.

An analogous bound can also be obtained for ℓ_1-regularized maxent in terms of the ℓ_∞ diameter of the feature space (relaxing the requirement of Corollary 5 that features be bounded in $[0,1]$):

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \frac{\|\boldsymbol{\lambda}^\star\|_1 D_\infty}{\sqrt{m}}\sqrt{2\ln(2n/\delta)} \ .$$

This bound increases with the ℓ_∞ diameter of the feature space and also grows slowly with the number of features. It provides some insight for when we expect ℓ_1 regularization to perform better than ℓ_2 regularization. For example, consider a scenario when the total number of features is large, but the best approximation of π can be derived from a small number of relevant features. Increasing the number of irrelevant features, we may keep $\|\boldsymbol{\lambda}^\star\|_1$, $\|\boldsymbol{\lambda}^\star\|_2$ and D_∞ fixed while increasing D_2 as $\Omega(\sqrt{n})$. The guarantee for ℓ_2-regularized maxent then grows as $\Omega(\sqrt{n})$ while the guarantee for ℓ_1-regularized maxent grows only as $\Omega(\sqrt{\ln n})$. Note, however, that in practice the distribution returned by ℓ_2-regularized maxent may perform better than indicated by this guarantee. For a comparison of ℓ_1 and ℓ_2^2 regularization in the context of logistic regression see Ng (2004).

5.4 Maxent with ℓ_2^2 Regularization

So far we have considered potentials that take the form of an indicator function or its smooth approximation. In this section we present a result for the ℓ_2^2 potential $\mathrm{U}_{\tilde{\pi}}^{(2)}$ of Section 4 and the corresponding conjugate $\mathrm{U}_{\tilde{\pi}}^{(2)*}$:

$$\mathrm{U}_{\tilde{\pi}}^{(2)}(\mathbf{u}) = \frac{\|\mathbf{u}\|_2^2}{2\alpha} \ , \qquad \mathrm{U}_{\tilde{\pi}}^{(2)*}(\boldsymbol{\lambda}) = \frac{\alpha\|\boldsymbol{\lambda}\|_2^2}{2} \ .$$

The potential $\mathrm{U}^{(2)}_{\tilde{\pi}}$ grows continuously with an increasing distance from empirical averages while the conjugate $\mathrm{U}^{(2)*}_{\tilde{\pi}}$ corresponds to ℓ_2^2 regularization.

In the case of ℓ_2^2-regularized maxent it is possible to derive guarantees on the expected performance in addition to probabilistic guarantees. However, these guarantees require an *a priori* bound on $\|\boldsymbol{\lambda}^\star\|_2$ and thus are not entirely uniform. Our expectation guarantees are analogous to those derived by Zhang (2005) for the conditional case. However, we are able to obtain a better multiplicative constant.

Note that we could derive expectation guarantees by simply applying Lemma 3.13 and taking the expectation over a random sample:

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \frac{\|\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]\|_2^2}{\alpha} + \alpha\|\boldsymbol{\lambda}^\star\|_2^2 \tag{19}$$
$$\mathrm{E}\big[\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}})\big] \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \frac{\mathrm{tr}\,\boldsymbol{\Sigma}}{\alpha m} + \alpha\|\boldsymbol{\lambda}^\star\|_2^2 \ .$$

Here, $\boldsymbol{\Sigma}$ is the covariance matrix of features with respect to π and tr denotes the trace of a matrix. We improve this guarantee by using Lemma 3.14 with $q_{\boldsymbol{\lambda}^\star}$ chosen to minimize $\mathrm{L}_\pi(\boldsymbol{\lambda}) + \mathrm{U}^{(2)*}_{\tilde{\pi}}(\boldsymbol{\lambda})$, and explicitly bounding $(\boldsymbol{\lambda}^\star - \hat{\boldsymbol{\lambda}}) \cdot (\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}])$ using a stability result similarly to Zhang (2005).

Lemma 12. *Let $\hat{\boldsymbol{\lambda}}$ minimize $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \alpha\|\boldsymbol{\lambda}\|_2^2/2$ where $\alpha > 0$. Then for every $q_{\boldsymbol{\lambda}^\star}$*

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \frac{\|\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]\|_2^2}{\alpha} + \frac{\alpha\|\boldsymbol{\lambda}^\star\|_2^2}{2} \ .$$

Proof of Lemma 12 is given in Appendix D. Lemma 12 improves on (19) in the leading constant of $\|\boldsymbol{\lambda}^\star\|_2^2$ which is $\alpha/2$ instead of α. Taking the expectation over a random sample and bounding the trace of $\boldsymbol{\Sigma}$ in terms of the ℓ_2 diameter (see Lemma 22 of Appendix C), we obtain an expectation guarantee. We can also use Lemma 10 to bound $\|\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]\|_2^2$ with high probability, and obtain a probabilistic guarantee. The two results are presented in Theorem 13 with the tradeoff between the guarantees controlled by the parameter s.

Theorem 13. *Let D_2 be the ℓ_2 diameter of $\mathbf{f}(\mathcal{X})$ and let $L_2, s > 0$. Let $\hat{\boldsymbol{\lambda}}$ minimize the ℓ_2^2-regularized log loss $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \alpha\|\boldsymbol{\lambda}\|_2^2/2$ with $\alpha = sD_2/(L_2\sqrt{m})$. Then*

$$\mathrm{E}\big[\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}})\big] \le \inf_{\|\boldsymbol{\lambda}^\star\|_2 \le L_2} \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \frac{L_2 D_2}{\sqrt{m}} \cdot \frac{s + s^{-1}}{2}$$

and if $\delta > 0$ then with probability at least $1 - \delta$

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \inf_{\|\boldsymbol{\lambda}^\star\|_2 \le L_2} \mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \frac{L_2 D_2}{\sqrt{m}} \cdot \frac{s + s^{-1}\big(1 + \sqrt{\ln(1/\delta)}\big)^2}{2} \ .$$

The bounds of Theorem 13 have properties similar to probabilistic guarantees of ℓ_2-regularized maxent. As mentioned earlier, they differ in the crucial fact that the norm $\|\boldsymbol{\lambda}^\star\|_2$ needs to be bounded *a priori* by a constant L_2. It is this constant rather than a possibly smaller norm $\|\boldsymbol{\lambda}^\star\|_2$ that enters the bound.

Note that bounds of this section generalize to arbitrary quadratic potentials $\mathrm{U}_{\tilde{\pi}}(\mathbf{u}) = \mathbf{u}^\top \mathbf{A}^{-1}\mathbf{u}/2$ and respective conjugates $\mathrm{U}^*_{\tilde{\pi}}(\boldsymbol{\lambda}) = \boldsymbol{\lambda}^\top \mathbf{A}\boldsymbol{\lambda}/2$ where $\mathbf{A}$ is a symmetric positive definite matrix. Applying the transformation

$$\mathbf{f}'(x) = \alpha^{1/2}\mathbf{A}^{-1/2}\mathbf{f}(x) \ , \qquad \boldsymbol{\lambda}' = \alpha^{-1/2}\mathbf{A}^{1/2}\boldsymbol{\lambda}$$

where $\mathbf{A}^{1/2}$ is the unique symmetric positive definite matrix such that $\mathbf{A}^{1/2}\mathbf{A}^{1/2} = \mathbf{A}$, the guarantees for quadratic-regularized maxent in terms of $\mathbf{f}(\mathcal{X})$ and $\boldsymbol{\lambda}^\star$ reduce to the guarantees for ℓ_2^2-regularized maxent in terms of $\mathbf{f}'(\mathcal{X})$ and $\boldsymbol{\lambda}^{\star\prime}$.

5.5 Maxent with ℓ_2 Regularization versus ℓ_2^2 Regularization

In the previous two sections we have seen that performance guarantees for maxent with ℓ_2 and ℓ_2^2 regularization differ whenever we require that β and α be fixed before running the algorithm. We now show that if all possible values of β and α are considered then the sets of models generated by the two maxent versions are the same.

Let $\Lambda^{(\sqrt{2}),\beta}$ and $\Lambda^{(2),\alpha}$ denote the respective solution sets for maxent with ℓ_2 and ℓ_2^2 regularization:

$$\Lambda^{(\sqrt{2}),\beta} = \arg\min_{\boldsymbol{\lambda}\in\mathbb{R}^n} \left[\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \beta\|\boldsymbol{\lambda}\|_2\right] \tag{20}$$

$$\Lambda^{(2),\alpha} = \arg\min_{\boldsymbol{\lambda}\in\mathbb{R}^n} \left[\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \alpha\|\boldsymbol{\lambda}\|_2^2/2\right] \quad . \tag{21}$$

If $\beta, \alpha > 0$ then $\Lambda^{(\sqrt{2}),\beta}$ and $\Lambda^{(2),\alpha}$ are non-empty because the objectives are lower semi-continuous and approach infinity as $\|\boldsymbol{\lambda}\|_2$ increases. For $\beta = 0$ and $\alpha = 0$, Equations (20) and (21) reduce to the basic maxent. Thus, $\Lambda^{(\sqrt{2}),0}$ and $\Lambda^{(2),0}$ contain the $\boldsymbol{\lambda}$'s for which $q_{\boldsymbol{\lambda}}[\mathbf{f}] = \tilde{\pi}[\mathbf{f}]$. This set will be empty if the basic maxent solutions are attained only in a limit.

Theorem 14. *Let* $\Lambda^{(\sqrt{2})} = \bigcup_{\beta\in[0,\infty]} \Lambda^{(\sqrt{2}),\beta}$ *and* $\Lambda^{(2)} = \bigcup_{\alpha\in[0,\infty]} \Lambda^{(2),\alpha}$. *Then* $\Lambda^{(\sqrt{2})} = \Lambda^{(2)}$.

Proof. First note that $\Lambda^{(\sqrt{2}),\infty} = \Lambda^{(2),\infty} = \{\mathbf{0}\}$. Next, we will show that $\Lambda^{(\sqrt{2})} \setminus \{\mathbf{0}\} = \Lambda^{(2)} \setminus \{\mathbf{0}\}$. Taking derivatives in Equations (20) and (21), we obtain that $\boldsymbol{\lambda} \in \Lambda^{(\sqrt{2}),\beta} \setminus \{\mathbf{0}\}$ if and only if

$$\boldsymbol{\lambda} \neq \mathbf{0} \quad \text{and} \quad \nabla\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \beta\boldsymbol{\lambda}/\|\boldsymbol{\lambda}\|_2 = 0 \quad .$$

Similarly, $\boldsymbol{\lambda} \in \Lambda^{(2),\alpha} \setminus \{\mathbf{0}\}$ if and only if

$$\boldsymbol{\lambda} \neq \mathbf{0} \quad \text{and} \quad \nabla\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \alpha\boldsymbol{\lambda} = 0 \quad .$$

Thus, any $\boldsymbol{\lambda} \in \Lambda^{(\sqrt{2}),\beta} \setminus \{\mathbf{0}\}$ is also in the set $\Lambda^{(2),\beta/\|\boldsymbol{\lambda}\|_2} \setminus \{\mathbf{0}\}$, and conversely any $\boldsymbol{\lambda} \in \Lambda^{(2),\alpha} \setminus \{\mathbf{0}\}$ is also in the set $\Lambda^{(\sqrt{2}),\alpha\|\boldsymbol{\lambda}\|_2} \setminus \{\mathbf{0}\}$. ■

The proof of Theorem 14 rests on the fact that the contours of regularization functions $\|\boldsymbol{\lambda}\|_2$ and $\|\boldsymbol{\lambda}\|_2^2$ coincide. We could easily extend the proof to include the equivalence of $\Lambda^{(\sqrt{2})}$, $\Lambda^{(2)}$ with the set of solutions to $\min\{\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) : \|\boldsymbol{\lambda}\|_2 \leq 1/\gamma\}$ where $\gamma \in [0,\infty]$. Similarly, one could show the equivalence of the solutions for regularizations $\beta\|\boldsymbol{\lambda}\|_1$, $\alpha\|\boldsymbol{\lambda}\|_1^2/2$ and $\mathrm{I}(\|\boldsymbol{\lambda}\|_1 \leq 1/\gamma)$.

The main implication of Theorem 14 is for maxent density estimation with model selection, for example, by minimization of held-out or cross-validated empirical error. In those cases, maxent versions with ℓ_2, ℓ_2^2 (and an ℓ_2-ball indicator) regularization yield the same solution. Thus, we prefer to use the computationally least intensive method. This will typically be ℓ_2^2-regularized maxent whose potential and regularization are smooth.

The solution sets $\Lambda^{(\sqrt{2}),\beta}$ and $\Lambda^{(2),\alpha}$ differ in their "sparsity" properties. We put the sparsity inside quotation marks because there are only two sparsity levels for ℓ_2 regularization: either all coordinates of $\boldsymbol{\lambda}$ remain zero under perturbations, or none of them. This is because the sole discontinuity of the gradient of the ℓ_2-regularized log loss is at $\boldsymbol{\lambda} = \mathbf{0}$. On the other hand, ℓ_2^2 regularization is smooth and therefore does not induce sparsity.

5.6 Maxent with $\ell_1 + \ell_2^2$ Regularization

In this section, we consider regularization that has both ℓ_1-style and ℓ_2^2-style terms. To simplify the discussion, we do not distinguish between coordinates and use a weighted sum of the ℓ_1 norm and the square of the ℓ_2 norm:

$$\mathrm{U}_{\tilde{\pi}}^{(1+2)*}(\boldsymbol{\lambda}) = \beta\|\boldsymbol{\lambda}\|_1 + \frac{\alpha\|\boldsymbol{\lambda}\|_2^2}{2} \ , \qquad \mathrm{U}_{\tilde{\pi}}^{(1+2)}(\mathbf{u}) = \sum_j \frac{\big||u_j| - \beta\big|_+^2}{2\alpha} \ .$$

Here α and β are positive constants, and $|x|_+ = \max\{0, x\}$ denotes the positive part of x. For a derivation of $\mathrm{U}_{\tilde{\pi}}^{(1+2)}$ see Appendix E.

For this type of regularization we are able to prove both probabilistic and expectation guarantees. Using similar techniques as in the previous sections we can derive, for example, the following theorem.

Theorem 15. *Let D_2, D_∞ be the ℓ_2 and ℓ_∞ diameters of $\mathbf{f}(\mathcal{X})$ respectively. Let $\delta, L_2 > 0$ and let $\hat{\boldsymbol{\lambda}}$ minimize $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \beta\|\boldsymbol{\lambda}\|_1 + \alpha\|\boldsymbol{\lambda}\|_2^2/2$ with $\alpha = (D_2 \min\{1/\sqrt{2}, \sqrt{m\delta}\})/(2L_2\sqrt{m})$ and $\beta = D_\infty\sqrt{\ln(2n/\delta)/(2m)}$. Then*

$$\mathrm{E}\big[\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}})\big] \le \inf_{\|\boldsymbol{\lambda}^\star\|_2 \le L_2} \left[\mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \frac{D_\infty\|\boldsymbol{\lambda}^\star\|_1}{\sqrt{m}}\sqrt{2\ln(2n/\delta)}\right] + \frac{D_2 L_2}{\sqrt{m}} \cdot \min\left\{\frac{1}{\sqrt{2}}, \sqrt{m\delta}\right\}$$

and with probability at least $1 - \delta$

$$\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) \le \inf_{\|\boldsymbol{\lambda}^\star\|_2 \le L_2} \left[\mathrm{L}_\pi(\boldsymbol{\lambda}^\star) + \frac{D_\infty\|\boldsymbol{\lambda}^\star\|_1}{\sqrt{m}}\sqrt{2\ln(2n/\delta)}\right] + \frac{D_2 L_2}{\sqrt{m}} \cdot \frac{1}{2}\min\left\{\frac{1}{\sqrt{2}}, \sqrt{m\delta}\right\} \ .$$

Proof. We only need to bound $\mathrm{U}_{\tilde{\pi}}^{(1+2)}(\tilde{\pi}[\mathbf{f}] - \pi[\mathbf{f}])$ and its expectation and use Lemma 3.14. By Hoeffding's inequality and the union bound, the potential is zero with probability at least $1 - \delta$, immediately yielding the second claim. Otherwise,

$$\mathrm{U}_{\tilde{\pi}}^{(1+2)}(\tilde{\pi}[\mathbf{f}] - \pi[\mathbf{f}]) \le \frac{\|\tilde{\pi}[\mathbf{f}] - \pi[\mathbf{f}]\|_2^2}{2\alpha} \le \frac{D_2^2}{2\alpha}$$

hence $\mathrm{E}\big[\mathrm{U}_{\tilde{\pi}}^{(1+2)}(\tilde{\pi}[\mathbf{f}] - \pi[\mathbf{f}])\big] \le (\delta D_2^2)/(2\alpha)$. On the other hand, we can bound the trace of the feature covariance matrix by Lemma 22 of Appendix C and obtain

$$\mathrm{E}\big[\mathrm{U}_{\tilde{\pi}}^{(1+2)}(\tilde{\pi}[\mathbf{f}] - \pi[\mathbf{f}])\big] \le \frac{\mathrm{E}\big[\|\tilde{\pi}[\mathbf{f}] - \pi[\mathbf{f}]\|_2^2\big]}{2\alpha} = \frac{\mathrm{tr}\,\boldsymbol{\Sigma}}{2m\alpha} \le \frac{D_2^2}{4m\alpha} \ .$$

Hence

$$\mathrm{E}\big[\mathrm{U}_{\tilde{\pi}}^{(1+2)}(\tilde{\pi}[\mathbf{f}] - \pi[\mathbf{f}])\big] \le \frac{D_2^2}{2m\alpha} \cdot \min\left\{\frac{1}{2}, m\delta\right\}$$

and the first claim follows. ■

Setting $\delta = s/m$, we bound the difference in performance between the maxent distribution and any Gibbs distribution of a bounded weight vector by $O\big(\big(D_\infty\|\boldsymbol{\lambda}^\star\|_1\sqrt{\ln(2mn/s)} + D_2 L_2\sqrt{s}\big)/\sqrt{m}\big)$. Now the constant s can be tuned to achieve the optimal tradeoff between $D_\infty\|\boldsymbol{\lambda}^\star\|_1$ and $D_2 L_2$. Notice that the sparsity inducing properties of ℓ_1 regularization are preserved in $\ell_1 + \ell_2^2$ regularization, because the partial derivatives of $\beta\|\boldsymbol{\lambda}\|_1 + \alpha\|\boldsymbol{\lambda}\|_2^2/2$ are discontinuous at zero.

5.7 Extensions to Other Regularization Types

Regularizations explored in previous sections are derived from ℓ_1 and ℓ_2 norms. However, performance guarantees easily extend to arbitrary norms using the corresponding concentration bounds and conjugacy relationships in the spirit of (Altun and Smola, 2006).

For instance, for an arbitrary norm $\|\cdot\|_{\mathcal{B}^*}$, the regularization function $\beta\|\boldsymbol{\lambda}\|_{\mathcal{B}^*}$ corresponds to the potential $\mathrm{I}_B(\mathbf{u})$ where $B = \{\|\mathbf{u}\|_{\mathcal{B}} \le \beta\}$ and $\|\cdot\|_{\mathcal{B}}$ is the dual norm of $\|\cdot\|_{\mathcal{B}^*}$. Similarly, for a norm $\|\cdot\|_{\mathcal{A}^*}$, the regularization function $\alpha\|\boldsymbol{\lambda}\|^2_{\mathcal{A}^*}/2$ corresponds to the potential $\|\mathbf{u}\|^2_{\mathcal{A}}/(2\alpha)$ where $\|\cdot\|_{\mathcal{A}}$ is the dual norm of $\|\cdot\|_{\mathcal{A}^*}$. For the combined regularization $\mathrm{U}^*_{\tilde{\pi}}(\boldsymbol{\lambda}) = \beta\|\boldsymbol{\lambda}\|_{\mathcal{B}^*} + \alpha\|\boldsymbol{\lambda}\|^2_{\mathcal{A}^*}/2$, we can perform a similar analysis as for $\ell_1 + \ell_2^2$ regularization using the bound

$$\mathrm{U}_{\tilde{\pi}}(\mathbf{u}) \le \min\{\mathrm{I}_B(\mathbf{u}), \|\mathbf{u}\|^2_{\mathcal{A}}/(2\alpha)\} \ .$$

Of course, the framework presented here is not limited to regularization functions derived from norms; however, the corresponding concentration bounds are typically less readily available.

6. Selective-Update Algorithm

In the previous section, we have discussed performance bounds of various types of regularization. Now we turn our attention to algorithms for solving generalized maxent problems. In the present and the following section, we propose two algorithms for generalized maxent with complete proofs of convergence. Our algorithms cover a wide class of potentials including the basic, box and ℓ_2^2 potential. The ℓ_2-ball potential $\mathrm{U}^{(\sqrt{2})}_{\tilde{\pi}}$ does not fall in this class, but we show that the corresponding maxent problem can be reduced and our algorithms can still be applied.

There are a number of algorithms for finding the basic maxent distribution, especially iterative scaling and its variants (Darroch and Ratcliff, 1972; Della Pietra et al., 1997). The **S**elective-**U**pdate algorithm for **M**aximu**M** **E**n**T**ropy (SUMMET) described in this section modifies one weight λ_j at a time, as explored by Collins, Schapire, and Singer (2002) in a similar setting. This style of coordinate-wise descent is convenient when working with a very large (or infinite) number of features. The original Darroch and Ratcliff algorithm also allows single-coordinate updates. Goodman (2002) observes that this leads to a much faster convergence than with the parallel version. However, updates are performed cyclically over all features, which renders the algorithm less practical with a large number of irrelevant features. Similarly, the sequential-update algorithm of Krishnapuram et al. (2005) requires a visitation schedule that updates each feature weight infinitely many times.

SUMMET differs since the weight to be updated is selected independently in each iteration. Thus, the features whose optimal weights are zero may never be updated. This approach is particularly useful in the context of ℓ_1-regularized maxent which often yields sparse solutions.

As explained in Section 4, the goal of the algorithm is to produce a sequence $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \dots$ maximizing the objective function Q in the limit. In this and the next section we assume that the potential U is *decomposable* as defined below:

Definition 16. A potential $\mathrm{U} : \mathbb{R}^n \to (-\infty, \infty]$ is called *decomposable* if it can be written as a sum of coordinate potentials $\mathrm{U}(\mathbf{u}) = \sum_j \mathrm{U}_j(u_j)$, each of which is a closed proper convex function bounded from below.

As a consequence of this definition, the conjugate potential U^* equals the sum of conjugate coordinate potentials U^*_j, by Equation (4), and $\mathrm{U}^*_j(0) = \sup_{u_j}[-\mathrm{U}_j(u_j)]$ is finite for all j.

```
Input: finite domain $\mathcal{X}$, default estimate $q_0$
       features $f_1, \dots, f_n$ where $f_j : \mathcal{X} \to [0,1]$, $f_j(\mathcal{X}) \neq \{0\}$ and $f_j(\mathcal{X}) \neq \{1\}$
       decomposable potential U where $\operatorname{dom} \mathrm{U}_j \cap [0,1] \neq \{0\}$ and $\operatorname{dom} \mathrm{U}_j \cap [0,1] \neq \{1\}$
Output: $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \dots$ maximizing $Q$
Let $\boldsymbol{\lambda}_1 = \mathbf{0}$
For $t = 1, 2, \dots$ :
  • let $(j^\star, \delta^\star) = \arg\max_{(j,\delta)} \Big[ -\ln\Big(1 + (e^\delta - 1) q_{\boldsymbol{\lambda}}[f_j]\Big) - \mathrm{U}^*_j(-\lambda_j - \delta) + \mathrm{U}^*_j(-\lambda_j) \Big]$
  • $\lambda_{t+1,j} = \begin{cases} \lambda_{t,j^\star} + \delta^\star & \text{if } j = j^\star \\ \lambda_{t,j} & \text{otherwise} \end{cases}$
```

Figure 2: **S**elective-**U**pdate algorithm for **M**aximu**M** **E**n**T**ropy (SUMMET).

Throughout this section we assume that values of features f_j lie in the interval $[0,1]$ and that features and coordinate potentials are non-degenerate in the sense that ranges $f_j(\mathcal{X})$ and intersections $\operatorname{dom} \mathrm{U}_j \cap [0,1]$ differ from $\{0\}$ and $\{1\}$. In Appendix G we show that a generalized maxent problem with a decomposable potential can always be reduced to the non-degenerate form.

Our algorithm works by iteratively adjusting the single weight λ_j that maximizes (an approximation of) the change in Q. To be more precise, suppose we add δ to λ_j. Let $\boldsymbol{\lambda}'$ be the resulting vector of weights, identical to $\boldsymbol{\lambda}$ except that $\lambda'_j = \lambda_j + \delta$. Then the change in the objective is

$$\begin{aligned} Q(\boldsymbol{\lambda}') - Q(\boldsymbol{\lambda}) &= -\ln Z_{\boldsymbol{\lambda}'} - \mathrm{U}^*(-\boldsymbol{\lambda}') + \ln Z_{\boldsymbol{\lambda}} + \mathrm{U}^*(-\boldsymbol{\lambda}) \\ &= -\ln\big(q_{\boldsymbol{\lambda}}\big[e^{\delta f_j}\big]\big) - \textstyle\sum_{j'} [\mathrm{U}^*_{j'}(-\lambda'_{j'}) - \mathrm{U}^*_{j'}(-\lambda_{j'})] \qquad (22) \\ &\geq -\ln\big(q_{\boldsymbol{\lambda}}\big[1 + (e^\delta - 1) f_j\big]\big) - \mathrm{U}^*_j(-\lambda_j - \delta) + \mathrm{U}^*_j(-\lambda_j) \qquad (23) \\ &= -\ln\big(1 + (e^\delta - 1) q_{\boldsymbol{\lambda}}[f_j]\big) - \mathrm{U}^*_j(-\lambda_j - \delta) + \mathrm{U}^*_j(-\lambda_j) \ . \qquad (24) \end{aligned}$$

Equation (22) uses

$$Z_{\boldsymbol{\lambda}'} = \sum_{x \in \mathcal{X}} q_0(x) e^{\boldsymbol{\lambda} \cdot \mathbf{f}(x) + \delta_j f_j(x)} = Z_{\boldsymbol{\lambda}} \sum_{x \in \mathcal{X}} q_{\boldsymbol{\lambda}}(x) e^{\delta_j f_j(x)} \ . \qquad (25)$$

Equation (23) is because $e^{\delta x} \leq 1 + (e^\delta - 1)x$ for $x \in [0,1]$ by convexity.

Let $F_j(\boldsymbol{\lambda}, \delta)$ denote the expression in (24):

$$F_j(\boldsymbol{\lambda}, \delta) = -\ln\big(1 + (e^\delta - 1) q_{\boldsymbol{\lambda}}[f_j]\big) - \mathrm{U}^*_j(-\lambda_j - \delta) + \mathrm{U}^*_j(-\lambda_j) \ .$$

Our algorithm, shown in Figure 2, on each iteration, maximizes this lower bound over all choices of (j, δ) and for the maximizing j adds the corresponding δ to λ_j. We assume that for each j the maximizing δ is finite. This will be the case if the potential and features are non-degenerate (see Appendix G). Note that $F_j(\boldsymbol{\lambda}, \delta)$ is strictly concave in δ so we can use any of a number of search methods to find the optimal δ.

Solving ℓ_1-Regularized Maxent. For maxent with box constraints (which subsumes the basic maxent), the optimizing δ can be derived explicitly. First note that

$$\begin{aligned} F^{(1)}_j(\boldsymbol{\lambda}, \delta) &= -\ln\big(1 + (e^\delta - 1) q_{\boldsymbol{\lambda}}[f_j]\big) - \mathrm{U}^{(1)*}_j(-\lambda_j - \delta) + \mathrm{U}^{(1)*}_j(-\lambda_j) \\ &= -\ln\big(1 + (e^\delta - 1) q_{\boldsymbol{\lambda}}[f_j]\big) + \delta \tilde{\pi}[f_j] - \beta_j(|\lambda_j + \delta| - |\lambda_j|) \end{aligned}$$

```
Input: finite domain $\mathcal{X}$, default estimate $q_0$
        examples $x_1, \ldots, x_m \in \mathcal{X}$
        features $f_1, \ldots, f_n$ where $f_j : \mathcal{X} \to [0,1]$, $f_j(\mathcal{X}) \neq \{0\}$ and $f_j(\mathcal{X}) \neq \{1\}$
        non-negative regularization parameters $\beta_1, \ldots, \beta_n$ where $\beta_j > 0$ if $\tilde{\pi}[f_j] \in \{0,1\}$
Output: $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \ldots$ minimizing $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \sum_j \beta_j |\lambda_j|$
Let $\boldsymbol{\lambda}_1 = \mathbf{0}$
For $t = 1, 2, \ldots$ :
  • $(j^\star, \delta^\star) = \arg\max_{(j,\delta)} \Big[ -\ln\Big(1 + (e^\delta - 1) q_t[f_j]\Big) + \tilde{\pi}[f_j]\delta - \beta_j(|\lambda_{t,j} + \delta| - |\lambda_{t,j}|) \Big]$
    for each $j$ it suffices to consider the following possibilities (whenever defined)
      $\delta_+ = \ln\left( \frac{(\tilde{\pi}[f_j] - \beta_j)(1 - q_t[f_j])}{(1 - \tilde{\pi}[f_j] + \beta_j) q_t[f_j]} \right)$ ,  $\delta_0 = -\lambda_{t,j}$ ,  $\delta_- = \ln\left( \frac{(\tilde{\pi}[f_j] + \beta_j)(1 - q_t[f_j])}{(1 - \tilde{\pi}[f_j] - \beta_j) q_t[f_j]} \right)$
    and choose $\delta_+$ if $\lambda_{t,j} + \delta_+ > 0$, $\delta_-$ if $\lambda_{t,j} + \delta_- < 0$, and $\delta_0$ otherwise
  • $\lambda_{t+1,j} = \begin{cases} \lambda_{t,j^\star} + \delta^\star & \text{if } j = j^\star \\ \lambda_{t,j} & \text{otherwise} \end{cases}$
```

Figure 3: Selective-update algorithm for ℓ_1-regularized maxent (ℓ_1-SUMMET).

since

$$\mathrm{U}_j^{(1)*}(-\mu_j) = \mathrm{U}_{\tilde{\pi},j}^{(1)*}(\mu_j) - \mu_j \tilde{\pi}[f_j] = \beta_j |\mu_j| - \mu_j \tilde{\pi}[f_j] \ .$$

The optimum δ can be obtained for each j via a simple case analysis on the sign of $\lambda_j + \delta$. In particular, using calculus, we see that we only need consider the possibility that $\delta = -\lambda_j$ or that δ is equal to

$$\ln\left(\frac{(\tilde{\pi}[f_j] - \beta_j)(1 - q_{\boldsymbol{\lambda}}[f_j])}{(1 - \tilde{\pi}[f_j] + \beta_j) q_{\boldsymbol{\lambda}}[f_j]} \right) \quad \text{or} \quad \ln\left(\frac{(\tilde{\pi}[f_j] + \beta_j)(1 - q_{\boldsymbol{\lambda}}[f_j])}{(1 - \tilde{\pi}[f_j] - \beta_j) q_{\boldsymbol{\lambda}}[f_j]} \right)$$

where the first and second of these can be valid only if $\lambda_j + \delta \geq 0$ and $\lambda_j + \delta \leq 0$, respectively. The complete algorithm, ℓ_1-SUMMET, is shown in Figure 3.

Solving ℓ_2-Regularized Maxent. The ℓ_2-ball potential $\mathrm{U}_{\tilde{\pi}}^{(\sqrt{2})}$ is not decomposable. In order to reduce ℓ_2-regularized maxent to maxent with a decomposable potential, we replace the constraint $\|\tilde{\pi}[\mathbf{f}] - p[\mathbf{f}]\|_2 \leq \beta$ by $\|\tilde{\pi}[\mathbf{f}] - p[\mathbf{f}]\|_2^2 \leq \beta^2$ which yields an equivalent primal:

$$\mathcal{P}' : \quad \min_{p \in \Delta} \mathrm{D}(p \parallel q_0) \text{ subject to } \|\tilde{\pi}[\mathbf{f}] - p[\mathbf{f}]\|_2^2 \leq \beta^2 \ .$$

If $\beta > 0$ then, by the Lagrange duality and Slater's conditions (Boyd and Vandenberghe, 2004), there exists $\mu \geq 0$ such that the solution of $\mathcal{P}'$ is the same as the solution of

$$\mathcal{P}'' : \quad \min_{p \in \Delta} \Big[\mathrm{D}(p \parallel q_0) + \mu \left(\|\tilde{\pi}[\mathbf{f}] - p[\mathbf{f}]\|_2^2 - \beta^2 \right) \Big] \ .$$

The sought-after μ is the one which maximizes the value of $\mathcal{P}''$. Since the value of $\mathcal{P}''$ is concave in μ, we can employ a range of search techniques to find the optimal μ, using SUMMET (or PLUMMET of the next section) with ℓ_2^2 regularization in each iteration.

6.1 Convergence

In order to prove convergence of SUMMET, we will measure its progress towards solving the primal and dual. One measure of progress is the difference between the primal evaluated at $q_{\boldsymbol{\lambda}}$ and the dual evaluated at $\boldsymbol{\lambda}$:

$$\begin{aligned}P(q_{\boldsymbol{\lambda}}) - Q(\boldsymbol{\lambda}) &= [\mathrm{D}(q_{\boldsymbol{\lambda}} \parallel q_0) + \mathrm{U}(q_{\boldsymbol{\lambda}}[\mathbf{f}])] - [-\ln Z_{\boldsymbol{\lambda}} - \mathrm{U}^*(-\boldsymbol{\lambda})] \\ &= q_{\boldsymbol{\lambda}}[\boldsymbol{\lambda}\cdot\mathbf{f} - \ln Z_{\boldsymbol{\lambda}}] + \mathrm{U}(q_{\boldsymbol{\lambda}}[\mathbf{f}]) + q_{\boldsymbol{\lambda}}[\ln Z_{\boldsymbol{\lambda}}] + \mathrm{U}^*(-\boldsymbol{\lambda}) \\ &= \mathrm{U}(q_{\boldsymbol{\lambda}}[\mathbf{f}]) + \mathrm{U}^*(-\boldsymbol{\lambda}) + \boldsymbol{\lambda}\cdot q_{\boldsymbol{\lambda}}[\mathbf{f}] \ .\end{aligned}$$

By Theorem 1, this difference is non-negative and equals zero exactly when $q_{\boldsymbol{\lambda}}$ solves primal and $\boldsymbol{\lambda}$ solves the dual.

For a decomposable potential, Fenchel's inequality in each coordinate implies that the difference is zero exactly when

$$\mathrm{U}_j(q_{\boldsymbol{\lambda}}[f_j]) + \mathrm{U}_j^*(-\lambda_j) + \lambda_j q_{\boldsymbol{\lambda}}[f_j] = 0$$

for all j. This characterization corresponds to the Kuhn-Tucker conditions (Rockafellar, 1970) in the case when coordinate potentials express equality and inequality constraints.

For many potentials of interest, including equality and inequality constraints, the difference between primal and dual may remain infinite throughout the computation. Therefore, we propose to use an *auxiliary function* as a surrogate for this difference. The auxiliary function is defined, somewhat non-standardly, as follows:

Definition 17. A function $A : \mathbb{R}^n \times \mathbb{R}^n \to (-\infty, \infty]$ is called an *auxiliary function* if

$$A(\boldsymbol{\lambda}, \mathbf{a}) = \mathrm{U}(\mathbf{a}) + \mathrm{U}^*(-\boldsymbol{\lambda}) + \boldsymbol{\lambda}\cdot\mathbf{a} + \mathrm{B}(\mathbf{a} \parallel q_{\boldsymbol{\lambda}}[\mathbf{f}])$$

where $\mathrm{B}(\cdot \parallel \cdot) : \mathbb{R}^n \times \mathbb{R}^n \to (-\infty, \infty]$ satisfies conditions (B1) and (B2).

The interpretation of an auxiliary function as a surrogate for the difference between primal and dual objectives is novel. Unlike previous applications of auxiliary functions (Della Pietra et al., 1997, 2001; Collins et al., 2002), we do not assume that $A(\boldsymbol{\lambda}, \mathbf{a})$ bounds a change in the dual objective and we also make no continuity assumptions. The reason for the former is technical: we need to allow a more flexible relationship between A and a change in the dual objective to accommodate algorithms both with single-coordinate and parallel updates. The absence of continuity assumptions is, however, crucial in order to allow arbitrary (decomposable) potentials. The continuity is replaced by the property (B2). On the other hand, our form of auxiliary function is more restrictive as the only flexibility is in choosing B, which is a function of $q_{\boldsymbol{\lambda}}[\mathbf{f}]$ rather than $q_{\boldsymbol{\lambda}}$.

The auxiliary function is always non-negative since $\mathrm{U}(\mathbf{a}) + \mathrm{U}^*(-\boldsymbol{\lambda}) \geq -\boldsymbol{\lambda}\cdot\mathbf{a}$ by Fenchel's inequality and hence $A(\boldsymbol{\lambda}, \mathbf{a}) \geq \mathrm{B}(\mathbf{a} \parallel q_{\boldsymbol{\lambda}}[\mathbf{f}]) \geq 0$. Moreover, if $A(\boldsymbol{\lambda}, \mathbf{a}) = 0$ then $q_{\boldsymbol{\lambda}}[\mathbf{f}] = \mathbf{a}$ and $A(\boldsymbol{\lambda}, \mathbf{a}) = P(q_{\boldsymbol{\lambda}}) - Q(\boldsymbol{\lambda}) = 0$, that is, by maxent duality, $q_{\boldsymbol{\lambda}}$ solves the primal and $\boldsymbol{\lambda}$ solves the dual.

It turns out, as we show in Lemma 19 below, that the optimality property generalizes to the case when $A(\boldsymbol{\lambda}_t, \mathbf{a}_t) \to 0$ provided that $Q(\boldsymbol{\lambda}_t)$ has a finite limit. In particular, it suffices to find a suitable sequence of $\mathbf{a}_t$'s for $\boldsymbol{\lambda}_t$'s produced by an algorithm to show its convergence. Note that the optimality in the limit trivially holds when $\boldsymbol{\lambda}_t$'s and $\mathbf{a}_t$'s come from a compact set, because $A(\hat{\boldsymbol{\lambda}}, \hat{\mathbf{a}}) = 0$ at a cluster point of $\{(\boldsymbol{\lambda}_t, \mathbf{a}_t)\}$ by the lower semi-continuity of U and U^*.

In a general case, we follow the technique used by Della Pietra, Della Pietra, and Lafferty (1997) for the basic maxent: we consider a cluster point $\hat{q}$ of $\{q_{\boldsymbol{\lambda}_t}\}$ and show that (i) $\hat{q}$ is primal feasible and (ii) the difference $P(\hat{q}) - Q(\boldsymbol{\lambda}_t)$ approaches zero. In the case of the basic maxent, $A(\boldsymbol{\lambda}, \mathbf{a}) = \mathrm{B}(\tilde{\pi}[\mathbf{f}] \parallel q_{\boldsymbol{\lambda}}[\mathbf{f}])$ whenever finite. Thus, (i) is obtained by (B2), and noting that $P(\hat{q}) - Q(\boldsymbol{\lambda}) = \mathrm{D}(\hat{q} \parallel q_{\boldsymbol{\lambda}})$ yields (ii). For a general potential, however, claims (i) and (ii) seem to require a novel approach. In both steps, we use decomposability and the technical Lemma 18 (proved in Appendix F). Thus compactness or decomposability seem to be crucial in the present approach.

Lemma 18. *Let* U_r *be a decomposable potential relative to a feasible point* r*. Let* $S = \operatorname{dom} \mathrm{U}_r = \{\mathbf{u} \in \mathbb{R}^n : \mathrm{U}_r(\mathbf{u}) < \infty\}$ *and* $T_c = \{\boldsymbol{\lambda} \in \mathbb{R}^n : \mathrm{U}_r^*(\boldsymbol{\lambda}) \le c\}$*. Then there exists* $\alpha_c \ge 0$ *such that* $\boldsymbol{\lambda} \cdot \mathbf{u} \le \alpha_c \|\mathbf{u}\|_1$ *for all* $\mathbf{u} \in S, \boldsymbol{\lambda} \in T_c$*.*

Lemma 19. *Let* $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \ldots \in \mathbb{R}^n$*,* $\mathbf{a}_1, \mathbf{a}_2, \ldots \in \mathbb{R}^n$ *be sequences such that* $Q(\boldsymbol{\lambda}_t)$ *has a finite limit and* $A(\boldsymbol{\lambda}_t, \mathbf{a}_t) \to 0$ *as* $t \to \infty$*. Then* $\lim_{t\to\infty} Q(\boldsymbol{\lambda}_t) = \sup_{\boldsymbol{\lambda}} Q(\boldsymbol{\lambda})$*.*

Proof. Let q_t denote $q_{\boldsymbol{\lambda}_t}$. Distributions q_t come from the compact set Δ, so we can choose a convergent subsequence. We index this subsequence by τ and denote its limit by $\hat{q}$. We assume that the subsequence was chosen in such a manner that values $A(\boldsymbol{\lambda}_\tau, \mathbf{a}_\tau)$ and $Q(\boldsymbol{\lambda}_\tau)$ are finite. We do this without loss of generality because limits of $A(\boldsymbol{\lambda}_\tau, \mathbf{a}_\tau)$ and $Q(\boldsymbol{\lambda}_\tau)$ are finite. We will show that $\lim_{\tau\to\infty} Q(\boldsymbol{\lambda}_\tau) = \sup_{\boldsymbol{\lambda}} Q(\boldsymbol{\lambda})$. The result will then follow since $\lim_{\tau\to\infty} Q(\boldsymbol{\lambda}_\tau) = \lim_{t\to\infty} Q(\boldsymbol{\lambda}_t)$.

As noted earlier, $A(\boldsymbol{\lambda}, \mathbf{a}) \ge \mathrm{B}(\mathbf{a} \parallel q_{\boldsymbol{\lambda}}[\mathbf{f}])$. Since $\mathrm{B}(\mathbf{a}_\tau \parallel q_\tau[\mathbf{f}])$ is non-negative and $A(\boldsymbol{\lambda}_\tau, \mathbf{a}_\tau) \to 0$, we obtain $\mathrm{B}(\mathbf{a}_\tau \parallel q_\tau[\mathbf{f}]) \to 0$. Thus, $\mathbf{a}_\tau \to \hat{q}[\mathbf{f}]$ by the property (B2). Rewriting A in terms of the potential and the conjugate potential relative to an arbitrary feasible point r (which exists by assumption), we obtain

$$A(\boldsymbol{\lambda}_\tau, \mathbf{a}_\tau) = \mathrm{U}_r(r[\mathbf{f}] - \mathbf{a}_\tau) + \mathrm{U}_r^*(\boldsymbol{\lambda}_\tau) - \boldsymbol{\lambda}_\tau \cdot (r[\mathbf{f}] - \mathbf{a}_\tau) + \mathrm{B}(\mathbf{a}_\tau \parallel q_\tau[\mathbf{f}]) \ . \tag{26}$$

Rearrange terms, noting that $A(\boldsymbol{\lambda}_\tau, \mathbf{a}_\tau) \to 0$ and $\mathrm{B}(\mathbf{a}_\tau \parallel q_\tau[\mathbf{f}]) \to 0$:

$$\mathrm{U}_r(r[\mathbf{f}] - \mathbf{a}_\tau) = -\mathrm{U}_r^*(\boldsymbol{\lambda}_\tau) + \boldsymbol{\lambda}_\tau \cdot (r[\mathbf{f}] - \mathbf{a}_\tau) + o(1) \ . \tag{27}$$

We use Equation (27) to prove first the feasibility and then the optimality of $\hat{q}$.

Feasibility. We bound the right hand site of Equation (27) and take limits to show that $\mathrm{U}_r(r[\mathbf{f}] - \hat{q}[\mathbf{f}])$ is also finite. The first term is bounded by Fenchel's inequality:

$$-\mathrm{U}_r^*(\boldsymbol{\lambda}_\tau) \le -\boldsymbol{\lambda}_\tau \cdot \mathbf{0} + \mathrm{U}_r(\mathbf{0}) = \mathrm{U}_r(\mathbf{0}) \ , \tag{28}$$

which is finite by the feasibility of r. In order to bound $\boldsymbol{\lambda}_\tau \cdot (r[\mathbf{f}] - \mathbf{a}_\tau)$, the second term of Equation (27), we use Lemma 18. First note that $r[\mathbf{f}] - \mathbf{a}_\tau$ is a feasible point of U_r for all τ by Equation (26) and the finiteness of $A(\boldsymbol{\lambda}_\tau, \mathbf{a}_\tau)$. Next, from Equation (11):

$$\mathrm{U}_r^*(\boldsymbol{\lambda}_\tau) = -Q(\boldsymbol{\lambda}_\tau) - \mathrm{D}(r \parallel q_\tau) + \mathrm{D}(r \parallel q_0) \ ,$$

which is bounded above by some constant c independent of τ because $-Q(\boldsymbol{\lambda}_\tau)$ has a finite limit and is thus bounded above, $-\mathrm{D}(r \parallel q_\tau)$ is non-positive, and $\mathrm{D}(r \parallel q_0)$ is a finite constant. Hence by Lemma 18

$$\boldsymbol{\lambda}_\tau \cdot (r[\mathbf{f}] - \mathbf{a}_\tau) \le \alpha_r \|r[\mathbf{f}] - \mathbf{a}_\tau\|_1 \tag{29}$$

for some constant α_r independent of τ. Plugging Equations (28) and (29) in Equation (27) and taking limits, we obtain by lower semi-continuity of U_r

$$\mathrm{U}_r(r[\mathbf{f}] - \hat{q}[\mathbf{f}]) \le \mathrm{U}_r(\mathbf{0}) + \alpha_r \|r[\mathbf{f}] - \hat{q}[\mathbf{f}]\|_1 \ .$$

Thus $\hat{q}$ is primal feasible.

Optimality. Since the foregoing holds for any primal feasible r, we can set $r = \hat{q}$ and obtain

$$\begin{aligned} \mathrm{U}_{\hat{q}}(\hat{q}[\mathbf{f}] - \mathbf{a}_\tau) &= -\mathrm{U}^*_{\hat{q}}(\boldsymbol{\lambda}_\tau) + \boldsymbol{\lambda}_\tau \cdot (\hat{q}[\mathbf{f}] - \mathbf{a}_\tau) + o(1) && (30)\\ &\le -\mathrm{U}^*_{\hat{q}}(\boldsymbol{\lambda}_\tau) + \alpha_{\hat{q}} \|\hat{q}[\mathbf{f}] - \mathbf{a}_\tau\|_1 + o(1) \ . && (31) \end{aligned}$$

Equation (30) follows from Equation (27). Equation (31) follows from Equation (29). Taking limits, we obtain

$$\mathrm{U}_{\hat{q}}(\mathbf{0}) \le \lim_{\tau\to\infty} \left[-\mathrm{U}^*_{\hat{q}}(\boldsymbol{\lambda}_\tau)\right] \ . \tag{32}$$

Now we are ready to show that $Q(\boldsymbol{\lambda}_\tau)$ maximizes the dual in the limit:

$$\begin{aligned} P(\hat{q}) &= \mathrm{D}(\hat{q} \parallel q_0) + \mathrm{U}_{\hat{q}}(\mathbf{0}) \\ &\le \mathrm{D}(\hat{q} \parallel q_0) + \lim_{\tau\to\infty} \left[-\mathrm{U}^*_{\hat{q}}(\boldsymbol{\lambda}_\tau)\right] && (33)\\ &= \lim_{\tau\to\infty} \left[\mathrm{D}(\hat{q} \parallel q_0) - \mathrm{D}(\hat{q} \parallel q_\tau) - \mathrm{U}^*_{\hat{q}}(\boldsymbol{\lambda}_\tau)\right] && (34)\\ &= \lim_{\tau\to\infty} Q(\boldsymbol{\lambda}_\tau) \ . && (35) \end{aligned}$$

Equation (33) follows from Equation (32). Equation (34) follows from the continuity of relative entropy since $q_\tau \to \hat{q}$. Equation (35) follows from Equation (11). Finally, combining Equations (33–35), we obtain by maxent duality that $\hat{q}$ minimizes the primal and λ_τ maximizes the dual as $\tau \to \infty$. ■

Theorem 20. SUMMET *produces a sequence* $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \ldots$ *for which*

$$\lim_{t\to\infty} Q(\boldsymbol{\lambda}_t) = \sup_{\boldsymbol{\lambda}} Q(\boldsymbol{\lambda}) \ .$$

Proof. It suffices to show that $Q(\boldsymbol{\lambda}_t)$ has a finite limit and present an auxiliary function A and a sequence $\mathbf{a}_1, \mathbf{a}_2, \ldots$ for which $A(\boldsymbol{\lambda}_t, \mathbf{a}_t) \to 0$.

Note that $Q(\boldsymbol{\lambda}_1) = Q(\mathbf{0}) = -\mathrm{U}^*(\mathbf{0})$ is finite by the decomposability of the potential, and Q is bounded above by the feasibility of the primal. Let $F_{t,j} = \max_\delta F_j(\boldsymbol{\lambda}_t, \delta)$. Note that $F_{t,j}$ is non-negative since $F_j(\boldsymbol{\lambda}_t, 0) = 0$. Since $F_{t,j}$ bounds change in the objective from below, the dual objective $Q(\boldsymbol{\lambda}_t)$ is non-decreasing and thus has a finite limit.

In each step

$$Q(\boldsymbol{\lambda}_{t+1}) - Q(\boldsymbol{\lambda}_t) \ge F_{t,j} \ge 0 \ .$$

Since Q has a finite limit, differences $Q(\boldsymbol{\lambda}_{t+1}) - Q(\boldsymbol{\lambda}_t)$ converge to zero and thus $F_{t,j} \to 0$. We use $F_{t,j}$ to define an auxiliary function. To begin, we rewrite $F_{t,j}$ using Fenchel's duality:

$$\begin{aligned} F_{t,j} &= \max_\delta \Big[-\ln(1 + (e^\delta - 1) q_t[f_j]) - \mathrm{U}^*_j(-\lambda_{t,j} - \delta) + \mathrm{U}^*_j(-\lambda_{t,j})\Big] \\ &= \max_\delta \Big[-\ln\big\{\big(1 - q_t[f_j]\big) e^{0\cdot\delta} + q_t[f_j] e^{1\cdot\delta}\big\} - \mathrm{U}'^*_j(-\delta)\Big] + \mathrm{U}^*_j(-\lambda_{t,j}) && (36)\\ &= \min_{a',a} \Big[\mathrm{D}\big((a', a) \parallel (1 - q_t[f_j], q_t[f_j])\big) + \mathrm{U}'_j(0 \cdot a' + 1 \cdot a)\Big] + \mathrm{U}^*_j(-\lambda_{t,j}) && (37)\\ &= \min_{0 \le a \le 1} \Big[\mathrm{D}(a \parallel q_t[f_j]) + \mathrm{U}_j(a) + \lambda_{t,j} \cdot a\Big] + \mathrm{U}^*_j(-\lambda_{t,j}) \ . && (38) \end{aligned}$$

```
Input: finite domain $\mathcal{X}$, default estimate $q_0$
       features $f_1,\ldots,f_n$ where $f_j : \mathcal{X} \to [0,1]$, $f_j(\mathcal{X}) \neq \{0\}$ and $\sum_j f_j(x) \leq 1$ for all $x \in \mathcal{X}$
       decomposable potential U where $\operatorname{dom} \mathrm{U}_j \cap [0,1] \neq \{0\}$
Output: $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \ldots$ maximizing $Q(\boldsymbol{\lambda})$
Let $\boldsymbol{\lambda}_1 = \mathbf{0}$
For $t = 1,2,\ldots$ :
  • for each $j$, let
        $\delta_j = \arg\max_{\delta} \left[-q_t[f_j](e^{\delta}-1) - \mathrm{U}^*_j(-\lambda_{t,j}-\delta) + \mathrm{U}^*_j(-\lambda_{t,j})\right]$
  • update $\boldsymbol{\lambda}_{t+1} = \boldsymbol{\lambda}_t + \boldsymbol{\delta}$
```

Figure 4: **P**aralle**L**-**U**pdate algorithm for **M**aximu**M** **E**n**T**ropy (PLUMMET).

In Equation (36), we rearranged terms inside the logarithm so they would take the form of a partition function. We write $\mathrm{U}'^*_j(u)$ for $\mathrm{U}^*_j(u-\lambda_{t,j})$. In Equation (37), we applied Theorem 1, noting that the conjugate of the log partition function is the relative entropy (see Section 3). The value of relative entropy $\mathrm{D}((a',a) \parallel (1-q_t[f_j], q_t[f_j]))$ is infinite whenever (a',a) is not a probability distribution, so it suffices to consider pairs where $0 \leq a \leq 1$ and $a' = 1-a$. In Equation (38), we use $\mathrm{D}(a \parallel q_t[f_j])$ as a shorthand for $\mathrm{D}((1-a,a) \parallel (1-q_t[f_j], q_t[f_j]))$. We use Equation (3) to convert U'_j into U_j:

$$\mathrm{U}'_j(0 \cdot a' + 1 \cdot a) = \mathrm{U}'_j(a) = \mathrm{U}_j(a) + \lambda_{t,j} \cdot a \ .$$

The minimum in Equation (38) is always attained because a comes from a compact set and the minimized expression is lower semi-continuous in a. We use $a_{t,j}$ to denote a value attaining this minimum. Thus

$$F_{t,j} = \mathrm{U}_j(a_{t,j}) + \mathrm{U}^*_j(-\lambda_{t,j}) + \lambda_{t,j} a_{t,j} + \mathrm{D}(a_{t,j} \parallel q_t[f_j]) \ .$$

Note that $\mathrm{D}(a \parallel b)$ satisfies conditions (B1) and (B2) hence the sum $\mathrm{B}(\mathbf{a} \parallel \mathbf{b}) = \sum_j \mathrm{D}(a_j \parallel b_j)$ also satisfies (B1) and (B2). We use this to derive the auxiliary function

$$A(\boldsymbol{\lambda}, \mathbf{a}) = \textstyle\sum_j \left[\mathrm{U}_j(a_j) + \mathrm{U}^*_j(-\lambda_j) + \lambda_j a_j + \mathrm{D}(a_j \parallel q_{\boldsymbol{\lambda}}[f_j])\right] \ .$$

Now $A(\boldsymbol{\lambda}_t, \mathbf{a}_t) = \sum_j F_{t,j} \to 0$, and the result follows by Lemma 19. ■

7. Parallel-Update Algorithm

Much of this paper has tried to be relevant to the case in which we are faced with a very large number of features. However, when the number of features is relatively small, it may be reasonable to maximize Q using an algorithm that updates all features simultaneously on every iteration. In this section, we describe a variant of generalized iterative scaling (Darroch and Ratcliff, 1972) applicable to generalized maxent with an arbitrary decomposable potential and prove its convergence. Note that gradient-based or Newton methods may be faster in practice.

Throughout this section, we make the assumption (without loss of generality) that, for all $x \in \mathcal{X}$, $f_j(x) \geq 0$ and $\sum_j f_j(x) \leq 1$ and features and coordinate potentials are non-degenerate in the sense that feature ranges $f_j(\mathcal{X})$ and intersections $\operatorname{dom} \mathrm{U}_j \cap [0,1]$ differ from $\{0\}$. Note that this differs from the notion of degeneracy in SUMMET.

```
Input: finite domain $\mathcal{X}$, default estimate $q_0$
       examples $x_1,\dots,x_m \in X$
       features $f_1,\dots,f_n$ where $f_j : \mathcal{X} \to [0,1]$, $f_j(\mathcal{X}) \neq \{0\}$ and $\sum_j f_j(x) \leq 1$ for all $x \in \mathcal{X}$
       nonnegative regularization parameters $\beta_1,\dots,\beta_n$ where $\beta_j > 0$ if $\tilde{\pi}[f_j] = 0$
Output: $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \dots$ minimizing $\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \sum_j \beta_j |\lambda_j|$
Let $\boldsymbol{\lambda}_1 = \mathbf{0}$
For $t = 1,2,\dots$ :
  • for each $j$, let
       $\delta_j = \arg\max_{\delta} \left[-q_t[f_j](e^{\delta}-1) + \delta\tilde{\pi}[f_j] - \beta_j(|\lambda_{t,j}+\delta| - |\lambda_{t,j}|)\right]$
       it suffices to consider the following possibilities (whenever defined)
          $\delta_+ = \ln\left(\frac{\tilde{\pi}[f_j]-\beta_j}{q_t[f_j]}\right)$ ,  $\delta_0 = -\lambda_{t,j}$ ,  $\delta_- = \ln\left(\frac{\tilde{\pi}[f_j]+\beta_j}{q_t[f_j]}\right)$
       and choose $\delta_+$ if $\lambda_{t,j}+\delta_+ > 0$, $\delta_-$ if $\lambda_{t,j}+\delta_- < 0$, and $\delta_0$ otherwise
  • update $\boldsymbol{\lambda}_{t+1} = \boldsymbol{\lambda}_t + \boldsymbol{\delta}$
```

Figure 5: Parallel-update algorithm for ℓ_1-regularized maxent (ℓ_1-PLUMMET).

Similarly to SUMMET, our **P**aralle**L**-**U**pdate algorithm for **M**aximu**M** **E**n**T**ropy (PLUMMET) is based on an approximation of the change in the objective function Q, in this case the following, where $\boldsymbol{\lambda}' = \boldsymbol{\lambda} + \boldsymbol{\delta}$:

$$\begin{aligned} Q(\boldsymbol{\lambda}') - Q(\boldsymbol{\lambda}) &= -\ln Z_{\boldsymbol{\lambda}'} - \mathrm{U}^*(-\boldsymbol{\lambda}') + \ln Z_{\boldsymbol{\lambda}} + \mathrm{U}^*(-\boldsymbol{\lambda}) \\ &= -\ln q_{\boldsymbol{\lambda}}\big[e^{\boldsymbol{\delta}\cdot\mathbf{f}}\big] - \mathrm{U}^*(-\boldsymbol{\lambda}-\boldsymbol{\delta}) + \mathrm{U}^*(-\boldsymbol{\lambda}) \qquad (39) \\ &\geq \sum_j \left[-q_{\boldsymbol{\lambda}}[f_j](e^{\delta_j}-1) - \mathrm{U}_j^*(-\lambda_j-\delta_j) + \mathrm{U}_j^*(-\lambda_j)\right] \ . \qquad (40) \end{aligned}$$

Equation (39) uses Equation (25). For Equation (40), note first that if $x_j \in \mathbb{R}$ and $p_j \geq 0$ with $\sum_j p_j \leq 1$ then

$$\exp\big(\textstyle\sum_j p_j x_j\big) - 1 \leq \sum_j p_j(e^{x_j}-1) \ .$$

(See Collins, Schapire, and Singer, 2002, for a proof.) Thus,

$$\begin{aligned} \ln q_{\boldsymbol{\lambda}}\big[\exp\big(\textstyle\sum_j \delta_j f_j\big)\big] &\leq \ln q_{\boldsymbol{\lambda}}\big[1 + \textstyle\sum_j f_j(e^{\delta_j}-1)\big] \\ &= \ln\big(1 + \textstyle\sum_j q_{\boldsymbol{\lambda}}[f_j](e^{\delta_j}-1)\big) \\ &\leq \textstyle\sum_j q_{\boldsymbol{\lambda}}[f_j](e^{\delta_j}-1) \end{aligned}$$

since $\ln(1+x) \leq x$ for all $x > -1$.

PLUMMET, shown in Figure 4, on each iteration, maximizes Equation (40) over all choices of the δ_j's. For the basic potential $\mathrm{U}^{(0)}$, this algorithm reduces to generalized iterative scaling of Darroch and Ratcliff (1972). For ℓ_1-style regularization, the maximizing $\boldsymbol{\delta}$ can be calculated explicitly (see algorithm ℓ_1-PLUMMET in Figure 5). Again, it turns out that all the components of the maximizing $\boldsymbol{\delta}$ are finite as long as features and potentials are non-degenerate (see Appendix G). As before, we can prove the convergence of PLUMMET, and thus also of ℓ_1-PLUMMET.

Theorem 21. PLUMMET *produces a sequence* $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \ldots$ *for which*

$$\lim_{t\to\infty} Q(\boldsymbol{\lambda}_t) = \sup_{\boldsymbol{\lambda}} Q(\boldsymbol{\lambda}) \ .$$

Proof. The proof mostly follows the same lines as the proof of Theorem 20. Here we sketch the main differences.

Let q_t denote $q_{\boldsymbol{\lambda}_t}$ and F_t denote the lower bound on the change in the objective:

$$F_t = \sup_{\boldsymbol{\delta}} \sum_j \Big[-q_t[f_j](e^{\delta_j} - 1) - \mathrm{U}_j^*(-\lambda_{t,j} - \delta_j) + \mathrm{U}_j^*(-\lambda_{t,j}) \Big] \ .$$

As before, $Q(\boldsymbol{\lambda}_t)$ has a finite limit and $F_t \to 0$. We can rewrite F_t using Fenchel's duality:

$$F_t = \sup_{\boldsymbol{\delta}} \sum_j \Big[-q_t[f_j](e^{\delta_j} - 1) - \mathrm{U}_j'^*(-\delta_j) \Big] + \mathrm{U}^*(-\boldsymbol{\lambda}_t) \tag{41}$$

$$= \inf_{\mathbf{a}\geq\mathbf{0}} \sum_j \Big[\widetilde{\mathrm{D}}(a_j \parallel q_t[f_j]) + \mathrm{U}_j'(a_j) \Big] + \mathrm{U}^*(-\boldsymbol{\lambda}_t) \tag{42}$$

$$= \inf_{\mathbf{a}\geq\mathbf{0}} \Big[\widetilde{\mathrm{D}}(\mathbf{a} \parallel q_t[\mathbf{f}]) + \mathrm{U}(\mathbf{a}) + \boldsymbol{\lambda}_t \cdot \mathbf{a} + \mathrm{U}^*(-\boldsymbol{\lambda}_t) \Big] \ . \tag{43}$$

In Equation (41) we write $\mathrm{U}_j'^*(u)$ for $\mathrm{U}_j^*(u - \lambda_{t,j})$. In Equation (42) we use Theorem 1, noting that the conjugate of $u_0(e^u - 1)$ is the unnormalized relative entropy. In Equation (43) we convert U_j' back into U_j and take the sum over j. Note that $\widetilde{\mathrm{D}}(\mathbf{a} \parallel q_t[\mathbf{f}])$ increases without bound if $\|\mathbf{a}\|_\infty \to \infty$ and, by Fenchel's inequality,

$$\mathrm{U}(\mathbf{a}) + \boldsymbol{\lambda}_t \cdot \mathbf{a} + \mathrm{U}^*(-\boldsymbol{\lambda}_t) \geq 0$$

so in Equation (43) it suffices to take an infimum over the $\mathbf{a}$'s of a bounded norm, that is, over a compact set. By lower semi-continuity we thus obtain that the infimum is attained at some point $\mathbf{a}_t$ and

$$F_t = \widetilde{\mathrm{D}}(\mathbf{a}_t \parallel q_t[\mathbf{f}]) + \mathrm{U}(\mathbf{a}_t) + \mathrm{U}^*(-\boldsymbol{\lambda}_t) + \boldsymbol{\lambda}_t \cdot \mathbf{a}_t \ .$$

Since $\widetilde{\mathrm{D}}(\mathbf{a} \parallel \mathbf{b})$ satisfies conditions (B1) and (B2), we obtain that

$$A(\boldsymbol{\lambda}, \mathbf{a}) = \widetilde{\mathrm{D}}(\mathbf{a} \parallel q_{\boldsymbol{\lambda}}[\mathbf{f}]) + \mathrm{U}(\mathbf{a}) + \mathrm{U}^*(-\boldsymbol{\lambda}) + \boldsymbol{\lambda} \cdot \mathbf{a}$$

is an auxiliary function. Noting that $A(\boldsymbol{\lambda}_t, \mathbf{a}_t) = F_t \to 0$ and using Lemma 19 yields the result. ■

8. Species Distribution Modeling Experiments

In this section we study how generalized maxent can be applied to the problem of modeling geographic distributions of species. This is a critical topic in ecology and conservation biology: to protect a threatened species, one first needs to know its environmental requirements, that is, its *ecological niche* (Hutchinson, 1957). A model of the ecological niche can further be used to predict the set of locations with sufficient conditions for the species to persist, that is, the *potential distribution* of the species (Anderson and Martínez-Meyer, 2004; Phillips et al., 2006), or the set of locations where conditions may become suitable under future climate conditions (Hannah et al., 2005). Ecological niche models are also useful for predicting the spread of invasive species and infectious diseases (Welk et al., 2002; Peterson and Shaw, 2003), as well as understanding ecological processes such as speciation (Graham et al., 2006).

As mentioned earlier, the input for species distribution modeling typically consists of a list of georeferenced occurrence localities as well as data on a number of environmental variables which have been measured or estimated across a geographic region of interest. The most basic goal is to predict which areas within the region are within the species' potential distribution. The potential distribution can be used to estimate the species' *realized distribution*, for example by removing areas where the species is known to be absent because of deforestation or other habitat destruction. Although a species' realized distribution may exhibit some spatial correlation, the potential distribution does not, so considering spatial correlation is not necessarily desirable during species distribution modeling.

It is often the case that only *presence* data is available indicating the occurrence of the species. Natural history museum and herbarium collections constitute the richest source of occurrence localities (Ponder et al., 2001; Stockwell and Peterson, 2002). Their collections typically have no information about the *failure* to observe the species at any given location; in addition, many locations have not been surveyed. In the lingo of machine learning, this means that we have only positive examples and no negative examples from which to learn. Moreover, the number of sightings (training examples) will often be very small by machine learning standards, for example, a hundred, ten, or even less. Thus, species distribution modeling is an example of a scientifically important problem which presents a challenging area for study by the machine learning community.

To explore the utility of generalized maxent and effects of regularization, we used ℓ_1-regularized maxent to model distributions of bird species, based on occurrence records in the North American Breeding Bird Survey (Sauer et al., 2001), an extensive data set consisting of thousands of occurrence localities for North American birds and used previously for species distribution modeling (Peterson, 2001). A preliminary version of these experiments and others was evaluated by Phillips, Dudík, and Schapire (2004).

In modeling species distributions from presence-only data, sample selection bias may hinder accurate prediction. Sample selection bias refers to the fact that observations are typically more likely in places that are easier to access, such as areas close to towns, roads, airports, or waterways. The impact of sample selection bias on maxent models, and various ways of coping with it are explored by Dudík, Schapire, and Phillips (2005). Here, we assume that the bias is not significant.

A comprehensive comparison of maxent and other species distribution modeling techniques was carried out by Elith et al. (2006) on a different data set than analyzed here. In that comparison, maxent is in the group of the best-performing methods. Here, we do not perform comparison with other approaches. We use species modeling as a setting to explore various aspects of ℓ_1-regularized maxent.

From the North American Breeding Bird Survey, we selected four species with a varying number of occurrence records: Hutton's Vireo (198 occurrences), Blue-headed Vireo (973 occurrences), Yellow-throated Vireo (1611 occurrences) and Loggerhead Shrike (1850 occurrences). The occurrence data of each species was divided into ten random partitions: in each partition, 50% of the occurrence localities were randomly selected for the training set, while the remaining 50% were set aside for testing. The environmental variables (coverages) use a North American grid with 0.2 degree square cells. We used seven coverages: elevation, aspect, slope, annual precipitation, number of wet days, average daily temperature and temperature range. The first three derive from a digital elevation model for North America (USGS, 2001), and the remaining four were interpolated from weather station readings (New et al., 1999). Each coverage is defined over a 386×286 grid, of which 58,065 points have data for all coverages. In addition to threshold features derived from

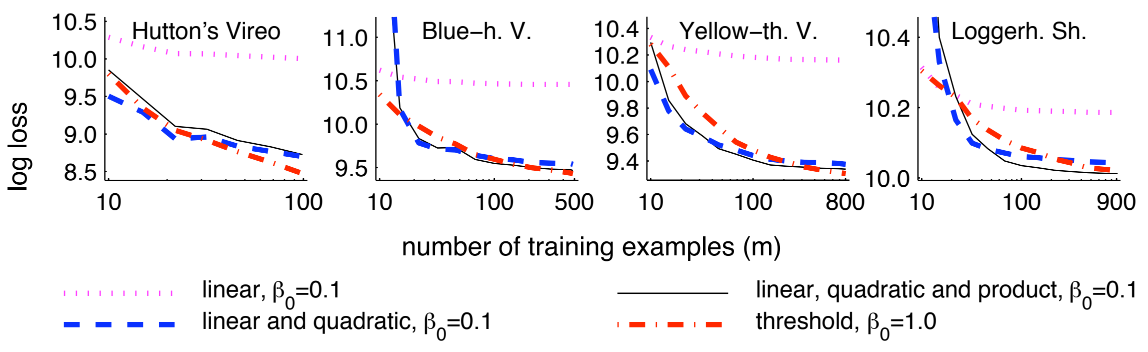


Figure 6: Learning curves. Log loss averaged over 10 partitions as a function of the number of training examples. Numbers of training examples are plotted on a logarithmic scale.

all environmental variables, we also used raw environmental variables (*linear* features), squares of environmental variables (*quadratic* features), and products of pairs of environmental variables (*product* features). Maxent with linear features finds the distribution of maximum entropy that matches empirical means of environmental variables; maxent with linear and quadratic features matches empirical means and variances; and maxent with linear, quadratic, and product features matches empirical means, variances, and covariances.

Recall that threshold features derived from a particular environmental variable are binary features equal to one if the variable is greater than a specified threshold and equal to zero otherwise. Formally, we consider a continuum of threshold features for each variable. In practice, it suffices to consider a single threshold between each pair of consecutive values appearing in the sample space; thus, in our data set we consider up to 58,064 threshold features for each variable. Given enough data, threshold features across all variables can model arbitrary additive responses in the exponent of the Gibbs distribution. Because of their expressivity, we expect that the danger of overfitting will be the most severe and regularization necessary.

In our experiments, we used ℓ_1-SUMMET of Section 6. All features are scaled to the interval $[0,1]$. Motivated by Corollary 7, we reduced the β_j's to a single regularization parameter β_0 by using $\beta_j = \beta_0 \tilde{\sigma}[f_j]/\sqrt{m}$. According to the bounds of Section 5.2, we expect that β_0 will depend on the number and complexity of features. Therefore, we expect that different values of β_0 will be optimal for different combinations of the feature types.

On each training set, we ran maxent with four different subsets of the feature types: linear (L); linear and quadratic (LQ); linear, quadratic and product (LQP); and threshold (T). We ran two types of experiments. First, we ran maxent on increasing subsets of the training data and evaluated log loss on the test data. We took an average over ten partitions and plotted the log loss as a function of the number of training examples. These plots are referred to as learning curves. Second, we also varied the regularization parameter β_0 and plotted the log loss for fixed numbers of training examples as functions of β_0. These curves are referred to as sensitivity curves.

In addition to these curves, we show how Gibbs distributions returned by maxent can be interpreted in terms of contribution of individual environmental variables to the exponent. The corresponding plots are called feature profiles. We give examples of feature profiles returned by maxent with and without regularization.

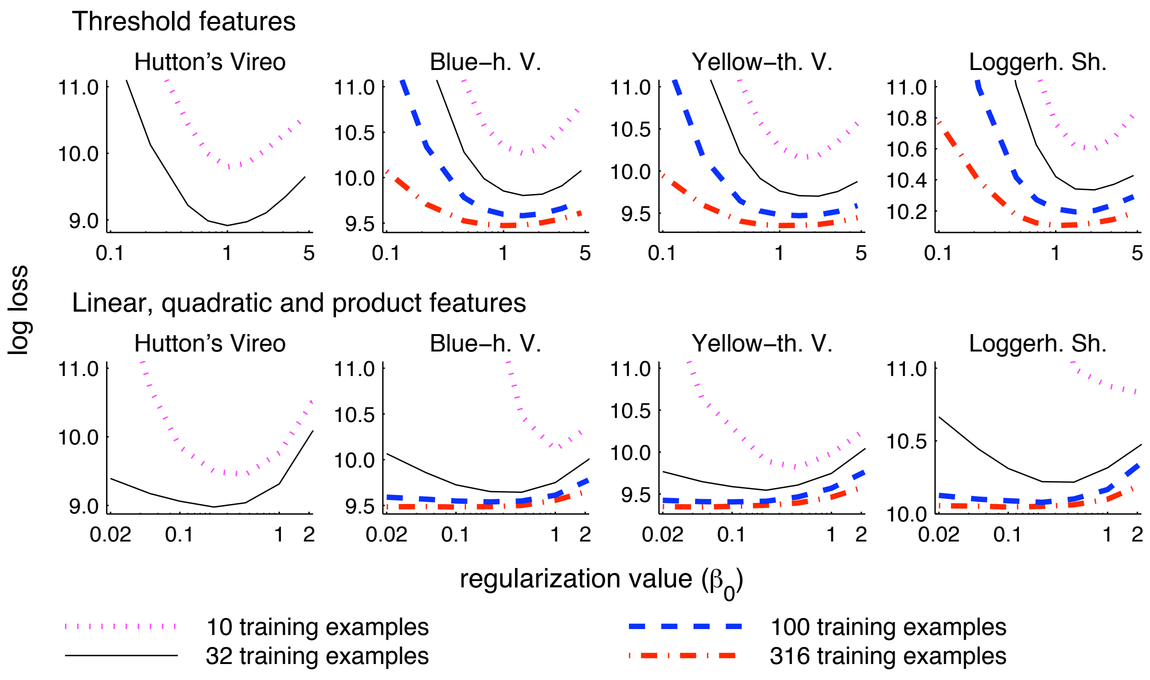


Figure 7: Sensitivity curves. Log loss averaged over 10 partitions as a function of β_0 for a varying number of training examples. For a fixed value of β_0, maxent finds better solutions (with smaller log loss) as the number of examples grows. Values of β_0 are plotted on a log scale.

Figure 6 shows learning curves for the four studied species. We set $\beta_0 = 0.1$ in L, LQ and LQP runs and $\beta_0 = 1.0$ in T runs. This choice is justified by the sensitivity curve experiments described below. In all cases, the performance improves as more samples become available. This is especially striking in the case of threshold features. In the absence of regularization, maxent would exactly fit the training data with delta functions around sample values of the environmental variables which would result in severe overfitting even when the number of training examples is large. As the learning curves show, regularized maxent does not exhibit this behavior.

Note the heavy overfitting of LQ and LQP features on the smallest sample sizes of Blue-headed Vireo and Loggerhead Shrike. A more detailed analysis of the sensitivity curves suggests that this overfitting could be alleviated by using larger values of β_0, resulting in curves qualitatively similar to those of other species. Similarly, performance of linear features, especially for larger feature sizes, could be somewhat improved using smaller regularization values.

Figure 7 shows the sensitivity of maxent to the regularization value β_0 for LQP and T versions of maxent. Results for L and LQ versions are similar to those for the LQP version. Note the remarkably consistent minimum at $\beta_0 \approx 1.0$ for threshold feature curves across different species, especially for larger sample sizes. It suggests that for the purposes of ℓ_1 regularization, $\tilde{\sigma}[f_j]/\sqrt{m}$ are good estimates of $|\tilde{\pi}[f_j] - \pi[f_j]|$ for threshold features. For LQP runs, the minima are much less pronounced as the number of samples increases and do not appear at the same value of β_0 across different species nor for different sizes of the same species. Benefits of regularization in LQP runs

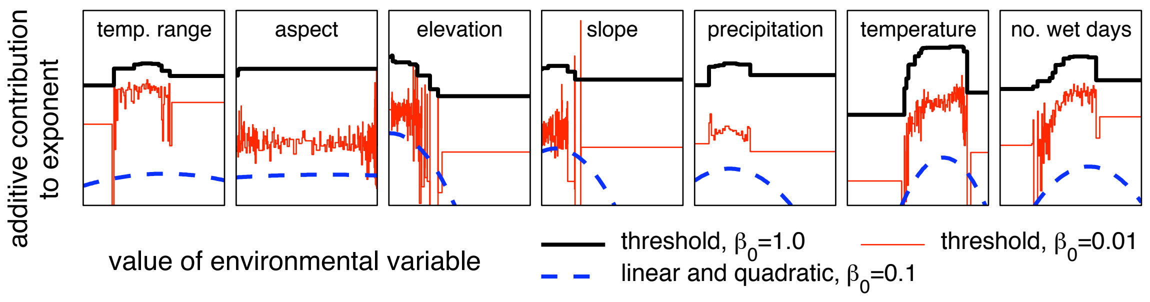


Figure 8: Feature profiles learned on the first partition of the Yellow-throated Vireo. For every environmental variable, its additive contribution to the exponent of the Gibbs distribution is given as a function of its value. Profiles have been shifted for clarity. This corresponds to adding a constant in the exponent, which has no effect on the resulting models since constants in the exponent cancel out with the normalization factor.

diminish as the number of training examples increases (this is even more so for LQ and L runs, not presented here). One possible explanation is that the relatively small number of features (compared with threshold features) prevents overfitting for large training sets.

To derive feature profiles, recall that maxent with a uniform default distribution returns the Gibbs distribution $q_{\boldsymbol{\lambda}}(x) = e^{\boldsymbol{\lambda}\cdot\mathbf{f}(x)}/Z_{\boldsymbol{\lambda}}$ minimizing the regularized log loss. For L, LQ, and T runs, the exponent is additive in contributions of individual environmental variables. Plotting this contribution as a function of the corresponding environmental variable we obtain feature profiles for the respective variables. Note that adding a constant to a profile has no impact on the resulting distribution as constants in the exponent cancel out with $Z_{\boldsymbol{\lambda}}$. For L models profiles are linear functions, for LQ models profiles are quadratic functions, and for T models profiles can be arbitrary piecewise constant functions. These profiles provide an easier to understand characterization of the distribution than the vector $\boldsymbol{\lambda}$.

Figure 8 shows feature profiles for an LQ run on the first partition of the Yellow-throated Vireo and two T runs with different values of β_0. The value of $\beta_0 = 0.01$ only prevents components of $\boldsymbol{\lambda}$ from becoming extremely large, but it does little to prevent heavy overfitting with numerous peaks capturing single training examples. Raising β_0 to 1.0 completely eliminates these peaks. This is especially prominent for the aspect variable where the regularized T as well as the LQ model show no dependence while the insufficiently regularized T model overfits heavily. Note the rough agreement between LQ profiles and regularized T profiles. Peaks in these profiles can be interpreted as intervals of environmental conditions favored by a species.[3]

3. Such interpretations should be made with caution as the objective of maxent is based solely on the predictive performance. In the extreme case, consider two identical environmental variables, only one of which has a causal effect on the species. Maxent has no knowledge which of the two variables is truly relevant, and may easily pick the wrong one, leaving the profile of the relevant one flat. Thus, interpretability is affected by correlations between variables.

9. Conclusion and Future Research Directions

The maximum entropy principle is a widely used method of density estimation. When the number of features is large, overfitting needs to be prevented by measures such as feature selection, regularization, discounting, or introduction of priors. In this work, we have provided a unified and complete account of maxent with generalized regularization. We have proved general performance guarantees and proposed versions of iterative scaling that incorporate regularization.

We have carried out analysis of several regularization types and presented scenarios in which these regularizations may be useful. In our experiments, we have shown how the principled ℓ_1 regularization facilitates learning. Further empirical study is needed to verify whether the theory derived for other regularization types corresponds to their performance. Generalizing the present analysis could help design task-specific regularization functions based on some prior information (for example properties of the feature space such as diameters with respect to various norms). Note that the quality of a regularization function can be assessed from two different perspectives: performance over test data and running time. The tradeoff between statistical guarantees and computational efficiency remains open for future research. In particular, convergence rates of algorithms presented in this paper are not known.

We have explored one direction of generalizing maxent: replacing equality constraints by an arbitrary convex potential in the primal or, equivalently, adding a convex regularization term to the maximum likelihood estimation in the dual. An alternative line of generalizations arises by replacing relative entropy in the primal objective by an arbitrary Bregman or Csiszár divergence along the lines of Altun and Smola (2006), and Collins, Schapire, and Singer (2002). Modified duality results and modified algorithms apply in the new setting, but performance guarantees do not directly translate to the case when divergences are derived from samples. Divergences of this kind are used in many cases of interest such as logistic regression (a conditional version of maxent) and boosting. In future work, we would like to generalize performance guarantees to these settings.

Finally, we have demonstrated the utility of generalized maxent in a novel application to species distribution modeling. We believe it is a scientifically important area that deserves the attention of the machine learning community while presenting some interesting challenges. Even though maxent fits the problem of species distribution modeling cleanly and effectively, there are many other techniques that could be used such as Markov random fields or mixture models. We leave the question of alternative machine learning approaches to species distribution modeling open for future research.

Acknowledgments

R. Schapire and M. Dudík received support through NSF grant CCR-0325463. We would like to thank anonymous referees for numerous suggestions that helped improve the quality of the paper.

Appendix A. Proof of Corollary 7

Proof of Corollary 7. Let

$$\beta'_j = \sqrt{\frac{\ln(4n/\delta)}{3m}} \cdot \sqrt{6\sigma^2[f_j] + \frac{\ln(4n/\delta)}{3m}} + \frac{\ln(4n/\delta)}{3m} \quad .$$

We will show that $|\tilde{\pi}[f_j]-\pi[f_j]| > \beta'_j$ with probability at most $\delta/2n$, and also $\beta'_j \geq \beta_j$ with probability at most $\delta/2n$. Then by the union bound, we obtain that

$$|\tilde{\pi}[f_j]-\pi[f_j]| \leq \beta'_j \leq \beta_j$$

for all j with probability at least $1-\delta$.

Consider a fixed j and let $\varepsilon = \ln(4n/\delta)/3m$. Thus,

$$\begin{aligned}
\beta'_j &= \sqrt{\varepsilon}\left(\sqrt{6\sigma^2[f_j]+\varepsilon}+\sqrt{\varepsilon}\right)\\
\beta_j &= \sqrt{6\varepsilon}\sqrt{\tilde{\sigma}^2[f_j]+\sqrt{\frac{\ln(2n/\delta)}{2m}}+\frac{\varepsilon}{6}}+\varepsilon\\
&= \sqrt{\varepsilon}\left(\sqrt{6\left[\tilde{\sigma}^2[f_j]+\sqrt{\ln(2n/\delta)/(2m)}\right]+\varepsilon}+\sqrt{\varepsilon}\right) \quad .
\end{aligned}$$

By Bernstein's inequality (Bernstein, 1946)

$$\begin{aligned}
\Pr\left[|\tilde{\pi}[f_j]-\pi[f_j]| > \beta'_j\right] &\leq 2\exp\left\{-\frac{3m\beta'^2_j}{6\sigma^2[f_j]+2\beta'_j}\right\}\\
&= 2\exp\left\{-\frac{3m\varepsilon\left(6\sigma^2[f_j]+\varepsilon+2\sqrt{\varepsilon}\sqrt{6\sigma^2[f_j]+\varepsilon}+\varepsilon\right)}{6\sigma^2[f_j]+2\sqrt{\varepsilon}\sqrt{6\sigma^2[f_j]+\varepsilon}+2\varepsilon}\right\}\\
&= 2\exp\{-3m\varepsilon\} = 2\exp\{-\ln(4n/\delta)\} = \delta/2n \quad .
\end{aligned}$$

To bound the probability that $\beta'_j \geq \beta_j$, it suffices to bound the probability of

$$\sigma^2[f_j] \geq \tilde{\sigma}^2[f_j]+\sqrt{\frac{\ln(2n/\delta)}{2m}} \quad .$$

We will use McDiarmid's inequality (McDiarmid, 1989) for the function

$$s(y_1,y_2,\ldots,y_m) = \frac{\sum_{i=1}^m y_i^2}{m-1} - \frac{\left(\sum_{i=1}^m y_i\right)^2}{m(m-1)} \quad .$$

Note that $\tilde{\sigma}^2[f_j] = s(f_j(x_1),f_j(x_2),\ldots,f_j(x_m))$ and $\mathrm{E}\left[\tilde{\sigma}^2[f_j]\right] = \sigma^2[f_j]$. By a simple case analysis,

$$\sup_{y_1,\ldots,y_m,y'_i\in[0,1]} |s(y_1,\ldots,y_m)-s(y_1,\ldots,y_{i-1},y'_i,y_{i+1},\ldots,y_m)| \leq \frac{1}{m}$$

for all i. Thus,

$$\begin{aligned}
\Pr\left[\sigma^2[f_j] \geq \tilde{\sigma}^2[f_j]+\sqrt{\frac{\ln(2n/\delta)}{2m}}\right] &\leq \exp\left\{\frac{-2\cdot\left[\ln(2n/\delta)/2m\right]}{m\cdot(1/m)^2}\right\}\\
&= \exp\{-\ln(2n/\delta)\} = \delta/2n \quad .
\end{aligned}$$

Hence also $\beta'_j \geq \beta_j$ with probability at most $\delta/2n$. ■

Appendix B. Derivation of $\mathrm{U}_{\tilde{\pi}}^{(\approx 1)}$

It suffices to derive a single coordinate potential $\mathrm{U}_{\tilde{\pi},j}^{(\approx 1)}$:

$$\begin{aligned}
\mathrm{U}_{\tilde{\pi},j}^{(\approx 1)}(u_j) &= \sup_{\lambda_j}\left[u_j\lambda_j - \alpha_j\beta_j \ln\left(\frac{e^{\lambda_j/\alpha_j}+e^{-\lambda_j/\alpha_j}}{2}\right)\right] \\
&= \alpha_j\beta_j \sup_{\lambda_j}\left[u_j\cdot\frac{\lambda_j}{\alpha_j\beta_j} - \ln\left(\frac{1}{2}\exp\left\{\beta_j\cdot\frac{\lambda_j}{\alpha_j\beta_j}\right\}+\frac{1}{2}\exp\left\{-\beta_j\cdot\frac{\lambda_j}{\alpha_j\beta_j}\right\}\right)\right] \\
&= \alpha_j\beta_j \sup_{\lambda'_j:=\lambda_j/\alpha_j\beta_j}\left[u_j\lambda'_j - \ln\left(\frac{1}{2}e^{\beta_j\lambda'_j}+\frac{1}{2}e^{-\beta_j\lambda'_j}\right)\right] \qquad (44) \\
&= \alpha_j\beta_j \mathrm{D}\left(\left(\frac{1+u_j/\beta_j}{2},\frac{1-u_j/\beta_j}{2}\right)\,\middle\|\,\left(\frac{1}{2},\frac{1}{2}\right)\right) \qquad (45)
\end{aligned}$$

Equation (44) follows by change of variables. Note that the supremum in Equation (44) takes form of a dual objective in a basic maxent over a two-sample space, say $\mathcal{X} = \{0,1\}$, with a single feature $f(0) = \beta_j, f(1) = -\beta_j$, and the empirical expectation $\tilde{\pi}[f] = u_j$. Thus, by maxent duality, the value of the supremum equals $\mathrm{D}(p \parallel (1/2,1/2))$, where p comes from a closure of the set of Gibbs distribution and $p[f] = u_j$. However, the only distribution on $\mathcal{X}$ that satisfies the expectation constraint is

$$p(0) = \frac{1+u_j/\beta_j}{2} \quad , \qquad p(1) = \frac{1-u_j/\beta_j}{2} \quad .$$

Thus, we obtain Equation (45).

Appendix C. Proof of Lemma 10

Before we proceed with the proof of Lemma 10, we show how the trace of the feature covariance matrix can be bounded in terms of the ℓ_2 diameter of the feature space.

Lemma 22. *Let $D_2 = \sup_{x,x'\in\mathcal{X}}\|\mathbf{f}(x)-\mathbf{f}(x')\|_2$ be the ℓ_2 diameter of $\mathbf{f}(\mathcal{X})$ and let $\mathbf{\Sigma} = \mathrm{E}\big[(\mathbf{f}(X)-\pi[\mathbf{f}])(\mathbf{f}(X)-\pi[\mathbf{f}])^\top\big]$, where X is distributed according to π, denote the feature covariance matrix. Then* $\operatorname{tr}\mathbf{\Sigma} \le D_2^2/2$.

Proof. Consider independent random variables X, X' distributed according to π. Let $\mathbf{f}, \mathbf{f}'$ denote the random variables $\mathbf{f}(X)$ and $\mathbf{f}(X')$. Then

$$\begin{aligned}
\mathrm{E}\big[\|\mathbf{f}-\mathbf{f}'\|_2^2\big] &= \mathrm{E}\big[\mathbf{f}\cdot\mathbf{f}\big] - 2\mathrm{E}\big[\mathbf{f}\big]\cdot\mathrm{E}\big[\mathbf{f}'\big] + \mathrm{E}\big[\mathbf{f}'\cdot\mathbf{f}'\big] \\
&= 2\mathrm{E}\big[\mathbf{f}\cdot\mathbf{f}\big] - 2\mathrm{E}\big[\mathbf{f}\big]\cdot\mathrm{E}\big[\mathbf{f}\big] \\
&= 2\textstyle\sum_j\big[\mathrm{E}\big[f_j^2\big] - (\mathrm{E}\big[f_j\big])^2\big] = 2\operatorname{tr}\mathbf{\Sigma} \quad .
\end{aligned}$$

Since $\|\mathbf{f}-\mathbf{f}'\|_2 \le D_2$, we obtain $\operatorname{tr}\mathbf{\Sigma} \le D_2^2/2$. ∎

Proof of Lemma 10. Consider independent samples $X_1,\ldots,X_m$ distributed according to π and the random variable $\mathbf{v}(X_1,\ldots,X_m) = \sum_i(\mathbf{f}(X_i)-\pi[\mathbf{f}]) = m(\tilde{\pi}[\mathbf{f}]-\pi[\mathbf{f}])$. We will bound $\mathrm{E}\big[\|\mathbf{v}\|_2\big]$ and use McDiarmid's inequality (McDiarmid, 1989) to show that

$$\Pr\Big[\|\mathbf{v}\|_2 - \mathrm{E}\big[\|\mathbf{v}\|_2\big] \ge D_2\sqrt{m\ln(1/\delta)/2}\Big] \le \delta \quad . \qquad (46)$$

By Jensen's inequality and Lemma 22, we obtain

$$\mathrm{E}\big[\|\mathbf{v}\|_2\big] \le \sqrt{\mathrm{E}\big[\|\mathbf{v}\|_2^2\big]} = \sqrt{m\,\mathrm{tr}\,\boldsymbol{\Sigma}} \le D_2\sqrt{m/2} \ .$$

Now, by the triangle inequality,

$$\begin{aligned}&\sup_{X_1,\ldots,X_m,X_i'} \Big| \big\|\mathbf{v}(X_1,\ldots,X_m)\big\|_2 - \big\|\mathbf{v}(X_1,\ldots,X_{i-1},X_i',X_{i+1},\ldots,X_m)\big\|_2 \Big| \\ &\qquad \le \sup_{X_i,X_i'} \|\mathbf{f}(X_i) - \mathbf{f}(X_i')\|_2 \le D_2 \ ,\end{aligned}$$

and Equation (46) follows by McDiarmid's inequality. ■

Appendix D. Proof of Lemma 12

Proof of Lemma 12. Let

$$\begin{aligned}\boldsymbol{\lambda}^{\star\star} &= \arg\min_{\boldsymbol{\lambda}} \left[\mathrm{L}_\pi(\boldsymbol{\lambda}) + \frac{\alpha\|\boldsymbol{\lambda}\|_2^2}{2} \right] \\ \hat{\boldsymbol{\lambda}} &= \arg\min_{\boldsymbol{\lambda}} \left[\mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \frac{\alpha\|\boldsymbol{\lambda}\|_2^2}{2} \right] \ .\end{aligned}$$

As the first step, we show that

$$\big\|\boldsymbol{\lambda}^{\star\star} - \hat{\boldsymbol{\lambda}}\big\|_2 \le \frac{\|\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]\|_2}{\alpha} \ . \tag{47}$$

Assume that $\boldsymbol{\lambda}^{\star\star} \ne \hat{\boldsymbol{\lambda}}$ (otherwise Equation (47) holds). Let $g(\boldsymbol{\lambda})$ denote $\ln Z_{\boldsymbol{\lambda}}$. This is the cumulant or log partition function of the family of Gibbs distributions. It is well known (and not difficult to show by calculus) that this function is convex in $\boldsymbol{\lambda}$. By the convexity of $g(\boldsymbol{\lambda})$ and $\alpha\|\boldsymbol{\lambda}\|_2^2/2$, the gradients of

$$\begin{aligned}\mathrm{L}_\pi(\boldsymbol{\lambda}) + \frac{\alpha\|\boldsymbol{\lambda}\|_2^2}{2} &= \ln Z_{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\cdot\pi[\mathbf{f}] + \frac{\alpha\|\boldsymbol{\lambda}\|_2^2}{2} \\ \mathrm{L}_{\tilde{\pi}}(\boldsymbol{\lambda}) + \frac{\alpha\|\boldsymbol{\lambda}\|_2^2}{2} &= \ln Z_{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\cdot\tilde{\pi}[\mathbf{f}] + \frac{\alpha\|\boldsymbol{\lambda}\|_2^2}{2}\end{aligned}$$

at their respective minima must equal zero:

$$\begin{aligned}\nabla g(\boldsymbol{\lambda}^{\star\star}) - \pi[\mathbf{f}] + \alpha\boldsymbol{\lambda}^{\star\star} &= 0 \\ \nabla g(\hat{\boldsymbol{\lambda}}) - \tilde{\pi}[\mathbf{f}] + \alpha\hat{\boldsymbol{\lambda}} &= 0 \ .\end{aligned}$$

Taking the difference yields

$$\alpha(\boldsymbol{\lambda}^{\star\star} - \hat{\boldsymbol{\lambda}}) = -(\nabla g(\boldsymbol{\lambda}^{\star\star}) - \nabla g(\hat{\boldsymbol{\lambda}})) + (\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]) \ .$$

Multiplying both sides by $(\boldsymbol{\lambda}^{\star\star} - \hat{\boldsymbol{\lambda}})$, we obtain

$$\begin{aligned}\alpha\|\boldsymbol{\lambda}^{\star\star} - \hat{\boldsymbol{\lambda}}\|_2^2 &= -(\boldsymbol{\lambda}^{\star\star} - \hat{\boldsymbol{\lambda}})\cdot(\nabla g(\boldsymbol{\lambda}^{\star\star}) - \nabla g(\hat{\boldsymbol{\lambda}})) + (\boldsymbol{\lambda}^{\star\star} - \hat{\boldsymbol{\lambda}})\cdot(\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]) \\ &\le (\boldsymbol{\lambda}^{\star\star} - \hat{\boldsymbol{\lambda}})\cdot(\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]) \qquad (48) \\ &\le \|\boldsymbol{\lambda}^{\star\star} - \hat{\boldsymbol{\lambda}}\|_2 \|\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]\|_2 \ . \qquad (49)\end{aligned}$$

Equation (48) follows because by convexity of $g(\boldsymbol{\lambda})$ for all $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2$

$$(\nabla g(\boldsymbol{\lambda}_2) - \nabla g(\boldsymbol{\lambda}_1)) \cdot (\boldsymbol{\lambda}_2 - \boldsymbol{\lambda}_1) \geq 0 \ .$$

Equation (49) follows by the Cauchy-Schwartz inequality. Dividing (49) by $\alpha \|\boldsymbol{\lambda}^{\star\star} - \hat{\boldsymbol{\lambda}}\|_2$ we obtain Equation (47). Now, by Lemma 3.13, the Cauchy-Schwartz inequality, Equation (47) and the optimality of $\boldsymbol{\lambda}^{\star\star}$ we obtain

$$\begin{aligned}
\mathrm{L}_\pi(\hat{\boldsymbol{\lambda}}) &\leq \mathrm{L}_\pi(\boldsymbol{\lambda}^{\star\star}) + (\boldsymbol{\lambda}^{\star\star} - \hat{\boldsymbol{\lambda}}) \cdot (\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]) + \mathrm{U}_{\tilde{\pi}}^{(2)*}(\boldsymbol{\lambda}^{\star\star}) - \mathrm{U}_{\tilde{\pi}}^{(2)*}(\hat{\boldsymbol{\lambda}}) \\
&\leq \mathrm{L}_\pi(\boldsymbol{\lambda}^{\star\star}) + \|\boldsymbol{\lambda}^{\star\star} - \hat{\boldsymbol{\lambda}}\|_2 \|\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]\|_2 + \frac{\alpha\|\boldsymbol{\lambda}^{\star\star}\|_2^2}{2} - \frac{\alpha\|\hat{\boldsymbol{\lambda}}\|_2^2}{2} \\
&\leq \mathrm{L}_\pi(\boldsymbol{\lambda}^{\star\star}) + \frac{\|\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]\|_2^2}{\alpha} + \frac{\alpha\|\boldsymbol{\lambda}^{\star\star}\|_2^2}{2} \\
&\leq \mathrm{L}_\pi(\boldsymbol{\lambda}^{\star}) + \frac{\|\pi[\mathbf{f}] - \tilde{\pi}[\mathbf{f}]\|_2^2}{\alpha} + \frac{\alpha\|\boldsymbol{\lambda}^{\star}\|_2^2}{2} \ .
\end{aligned}$$

■

Appendix E. Derivation of $\mathrm{U}_{\tilde{\pi}}^{(1+2)}$

It suffices to derive a single coordinate potential $\mathrm{U}_{\tilde{\pi},j}^{(1+2)}$:

$$\begin{aligned}
\mathrm{U}_{\tilde{\pi},j}^{(1+2)}(u_j) &= \sup_{\lambda_j} \left(u_j\lambda_j - \beta|\lambda_j| - \frac{\alpha\lambda_j^2}{2} \right) \\
&= \sup_{\lambda_j:\, u_j\lambda_j = |u_j||\lambda_j|} \left(\frac{\alpha}{2} \cdot |\lambda_j| \cdot \left[\frac{2\,(|u_j| - \beta)}{\alpha} - |\lambda_j| \right] \right) \ .
\end{aligned} \tag{50}$$

In Equation (50) we note that for each pair $\pm\lambda_j$, it suffices to consider the value whose sign agrees with u_j. Next, if $|u_j| \leq \beta$ then the bracketed expression is non-positive, hence the supremum is attained at $\lambda_j = 0$ and its value equals 0. For $|u_j| > \beta$, the supremum is attained when $|\lambda_j| = (|u_j| - \beta)/\alpha$, in which case its value equals $(|u_j| - \beta)^2/(2\alpha)$.

Appendix F. Proof of Lemma 18

We will first prove a single coordinate version of Lemma 18 and then turn to the general case.

Lemma 23. *Let $\psi : \mathbb{R} \to (-\infty, \infty]$ be a proper closed convex function. Let $S = \operatorname{dom}\psi = \{u \in \mathbb{R} : \psi(u) < \infty\}$ and $T_c = \{v \in \mathbb{R} : \psi^*(v) \leq c\}$. Then there exists $\alpha_c \geq 0$ such that $uv \leq \alpha_c|u|$ for all $u \in S, v \in T_c$.*

Proof. Inequality $uv \leq \alpha_c|u|$ holds for an arbitrary α_c if $u = 0$. We determine α_c separately for cases $u \in S_+ = S \cap (0, \infty)$ and $u \in S_- = S \cap (-\infty, 0)$ and choose the maximum.

Assume $S_+ \neq \emptyset$ and pick an arbitrary $u_+ \in S_+$. Then for any $v \in T_c$ by Fenchel's inequality

$$u_+v \leq \psi(u_+) + \psi^*(v) \leq \psi(u_+) + c$$

and thus

$$v \leq \frac{\psi(u_+) + c}{u_+} \ .$$

Now for any $u \in S_+$

$$uv \le u \cdot \frac{\psi(u_+) + c}{u_+} \le |u| \cdot \left| \frac{\psi(u_+) + c}{u_+} \right| \quad .$$

Similarly, if $S_- \neq \emptyset$ then we can choose an arbitrary $u_- \in S_-$ and obtain for all $u \in S_-$

$$uv \le |u| \cdot \left| \frac{\psi(u_-) + c}{u_-} \right| .$$

To complete the proof we choose

$$\alpha_c = \max \left\{ \left| \frac{\psi(u_+) + c}{u_+} \right|, \left| \frac{\psi(u_-) + c}{u_-} \right| \right\}$$

setting the respective terms to 0 if S_+ or S_- is empty. ■

Proof of Lemma 18. Assume that $\mathrm{U}_r(\mathbf{u}) < \infty$ and thus by decomposability $\mathrm{U}_{r,j}(u_j) < \infty$ for all j. Also assume that $\mathrm{U}_r^*(\boldsymbol{\lambda}) = \sum_j \mathrm{U}_{r,j}^*(\lambda_j) < c$. By Fenchel's inequality $\mathrm{U}_{r,j}^*(\lambda_j) \ge -\mathrm{U}_{r,j}(0)$ which is finite by the feasibility of r. Since the sum of $\mathrm{U}_{r,j}^*(\lambda_j)$ is bounded above by c and individual functions are bounded below by constants, they must also be bounded above by some constants c_j. By Lemma 23 applied to coordinate potentials, we obtain that $u_j\lambda_j \le \alpha_j|u_j|$ for some constants $\alpha_1, \ldots, \alpha_n$. The conclusion follows by taking $\alpha_c = \max_j \alpha_j$. ■

Appendix G. Ensuring Finite Updates

In this appendix, we discuss how to ensure that features and coordinate potentials are non-degenerate in SUMMET and PLUMMET, and show that non-degeneracy implies that updates in both algorithms are always finite.

G.1 Non-degeneracy in SUMMET

In SUMMET, we assume that $f_j(\mathcal{X}) \subseteq [0,1]$. In context of this algorithm, a feature f_j is degenerate if $f_j(\mathcal{X}) = \{0\}$ or $f_j(\mathcal{X}) = \{1\}$ and a coordinate potential U_j is degenerate if $\operatorname{dom}\mathrm{U}_j \cap [0,1] = \{0\}$ or $\operatorname{dom}\mathrm{U}_j \cap [0,1] = \{1\}$. In order to obtain non-degenerate features and coordinate potentials, it suffices to preprocess the sample space $\mathcal{X}$ and the feature set as follows:

1. For all j: if $\operatorname{dom}\mathrm{U}_j \cap [0,1] = \{0\}$ then $\mathcal{X} \leftarrow \{x \in \mathcal{X} : f_j(x) = 0\}$.
2. For all j: if $\operatorname{dom}\mathrm{U}_j \cap [0,1] = \{1\}$ then $\mathcal{X} \leftarrow \{x \in \mathcal{X} : f_j(x) = 1\}$.
3. For all j: if $f_j(x) = 0$ for all $x \in \mathcal{X}$ then remove feature f_j.
4. For all j: if $f_j(x) = 1$ for all $x \in \mathcal{X}$ then remove feature f_j.

Whenever U_j is degenerate, steps 1–2 guarantee that f_j will be eventually removed in steps 3–4. While f_j could be removed immediately in steps 1–2, note that steps 3–4 are still necessary since features may be degenerate even when potentials are not. Also note that steps 1–2 must precede steps 3–4 since restricting $\mathcal{X}$ may introduce new degenerate features.

The preprocessing described above yields an equivalent form of the primal. By restricting the sample space in steps 1–2, we effectively eliminate distributions that are nonzero outside the restricted sample set. Note that those distributions are infeasible because their feature means lie outside $\operatorname{dom}\mathrm{U}$. In steps 3–4, we simply remove constant terms of the potential function.

Theorem 24. *Let $\boldsymbol{\lambda}$ and $Q(\boldsymbol{\lambda})$ be finite and f_j, U_j non-degenerate. Then $F_j(\boldsymbol{\lambda},\delta)$ is maximized by a finite δ.*

Proof. We will show that $F_j(\boldsymbol{\lambda},\delta) \to -\infty$ if $\delta \to \pm\infty$. Thus, it suffices to consider δ from a compact interval and the result follows by upper semi-continuity of F_j. First, consider the case $\delta \to \infty$. Let r be an arbitrary feasible distribution. Rewrite $F_j(\boldsymbol{\lambda},\delta)$ as follows:

$$\begin{aligned}
F_j(\boldsymbol{\lambda},\delta) &= -\ln\bigl(1+(e^{\delta}-1)q_{\boldsymbol{\lambda}}[f_j]\bigr) - \mathrm{U}_j^*(-\lambda_j-\delta) + \mathrm{U}_j^*(-\lambda_j) \\
&= -\ln\left\{e^{\delta}\left[e^{-\delta}(1-q_{\boldsymbol{\lambda}}[f_j]) + q_{\boldsymbol{\lambda}}[f_j]\right]\right\} + \delta r[f_j] - \mathrm{U}_{r,j}^*(\lambda_j+\delta) + \mathrm{U}_{r,j}^*(\lambda_j) \\
&= -\ln\left[e^{-\delta}(1-q_{\boldsymbol{\lambda}}[f_j]) + q_{\boldsymbol{\lambda}}[f_j]\right] - \delta(1-r[f_j]) - \mathrm{U}_{r,j}^*(\lambda_j+\delta) + \mathrm{U}_{r,j}^*(\lambda_j). \qquad (51)
\end{aligned}$$

Suppose that $r[f_j] < 1$. Then $F_j(\boldsymbol{\lambda},\delta) \to -\infty$: the first term of (51) is bounded above by $-\ln(q_{\boldsymbol{\lambda}}[f_j])$ which is finite by non-degeneracy of f_j; the second term decreases without bound; the third term is bounded above by $\mathrm{U}_{r,j}(0)$ by Fenchel's inequality; and the fourth term is a finite constant because $Q(\boldsymbol{\lambda})$ is finite. In case $r[f_j] = 1$, the second term equals zero, but the third term decreases without bound because by non-degeneracy of U_j there exists $\varepsilon > 0$ such that $\mathrm{U}_{r,j}(\varepsilon) = \mathrm{U}_j(1-\varepsilon) < \infty$ and hence by Fenchel's inequality $-\mathrm{U}_{r,j}^*(\lambda_j+\delta) \le -(\lambda_j+\delta)\varepsilon + \mathrm{U}_{r,j}(\varepsilon)$.

Now consider $\delta \to -\infty$ and rewrite $F_j(\boldsymbol{\lambda},\delta)$ as follows:

$$F_j(\boldsymbol{\lambda},\delta) = -\ln\bigl((1-q_{\boldsymbol{\lambda}}[f_j]) + e^{\delta}q_{\boldsymbol{\lambda}}[f_j]\bigr) + \delta r[f_j] - \mathrm{U}_{r,j}^*(\lambda_j+\delta) + \mathrm{U}_{r,j}^*(\lambda_j)\ .$$

Assuming that $r[f_j] > 0$, the second term decreases without bound and the remaining terms are bounded above. If $r[f_j] = 0$ then the third term decreases without bound because by non-degeneracy of U_j there exists $\varepsilon > 0$ such that $\mathrm{U}_{r,j}(-\varepsilon) = \mathrm{U}_j(\varepsilon) < \infty$ and thus by Fenchel's inequality $-\mathrm{U}_{r,j}^*(\lambda_j+\delta) \le (\lambda_j+\delta)\varepsilon + \mathrm{U}_{r,j}(-\varepsilon)$. ■

Corollary 25. *Updates of* SUMMET *are always finite.*

Proof. We proceed by induction. In the first step, both $\boldsymbol{\lambda}_1$ and $Q(\boldsymbol{\lambda}_1)$ are finite (see proof of Theorem 20). Now suppose that in step t, $\boldsymbol{\lambda}_t$ and $Q(\boldsymbol{\lambda}_t)$ are finite. Then by Theorem 24, all considered coordinate updates will be finite, so $\boldsymbol{\lambda}_{t+1}$ will be finite too. Since $Q(\boldsymbol{\lambda}_{t+1}) \ge Q(\boldsymbol{\lambda}_t)$ and $Q(\boldsymbol{\lambda})$ is bounded above (see proof of Theorem 20), we obtain that $Q(\boldsymbol{\lambda}_{t+1})$ is finite. ■

G.2 Non-degeneracy in PLUMMET

In this case, we assume that $f_j(x) \ge 0$ and $\sum_j f_j(x) \le 1$ for all $x \in \mathcal{X}$. We call a feature f_j degenerate if $f_j(\mathcal{X}) = \{0\}$ and a coordinate potential U_j degenerate if $\mathrm{dom}\,\mathrm{U}_j \cap [0,1] = \{0\}$. To obtain non-degenerate features and coordinate potentials, it suffices to preprocess the sample space $\mathcal{X}$ and the feature set as follows:

1. For all j: if $\mathrm{dom}\,U_j \cap [0,1] = \{0\}$ then $\mathcal{X} \leftarrow \{x \in \mathcal{X} : f_j(x) = 0\}$.
2. For all j: if $f_j(x) = 0$ for all $x \in \mathcal{X}$ then remove feature f_j.

Similarly to SUMMET, this preprocessing derives an equivalent form of the primal. Using analogous reasoning as in Theorem 24, we show below that non-degeneracy implies finite updates in PLUMMET.

In each iteration of the algorithm we determine updates δ_j by maximizing

$$\begin{aligned} F_j(\boldsymbol{\lambda},\delta) &= -q_{\boldsymbol{\lambda}}[f_j](e^{\delta}-1) - \mathrm{U}_j^*(-\lambda_j-\delta) + \mathrm{U}_j^*(-\lambda_j) \\ &= -q_{\boldsymbol{\lambda}}[f_j](e^{\delta}-1) + \delta r[f_j] - \mathrm{U}_{r,j}^*(\lambda_j+\delta) + \mathrm{U}_{r,j}^*(\lambda_j) \ . \end{aligned}$$

It suffices to prove that $F_j(\boldsymbol{\lambda},\delta) \to -\infty$ if $\delta \to \pm\infty$ given that $Q(\boldsymbol{\lambda})$ and λ_j are finite and f_j, U_j are non-degenerate.

First, we rewrite F_j as follows:

$$F_j(\boldsymbol{\lambda},\delta) = -e^{\delta}\left[q_{\boldsymbol{\lambda}}[f_j] - e^{-\delta}q_{\boldsymbol{\lambda}}[f_j] - e^{-\delta}\delta r[f_j]\right] - \mathrm{U}_{r,j}^*(\lambda_j+\delta) + \mathrm{U}_{r,j}^*(\lambda_j) \ .$$

If $\delta \to \infty$ then the expression in the brackets approaches $q_{\boldsymbol{\lambda}}[f_j]$, which is positive by non-degeneracy of f_j. Thus the first term decreases without bound while the second and third terms are bounded from above. Next, rewrite F_j as

$$F_j(\boldsymbol{\lambda},\delta) = \delta\left[r[f_j] - \frac{e^{\delta}}{\delta}q_{\boldsymbol{\lambda}}[f_j] + \frac{1}{\delta}q_{\boldsymbol{\lambda}}[f_j]\right] - \mathrm{U}_{r,j}^*(\lambda_j+\delta) + \mathrm{U}_{r,j}^*(\lambda_j) \ .$$

If $\delta \to -\infty$ then the expression in the brackets approaches $r[f_j]$. Thus, if $r[f_j] > 0$ then the first term decreases without bound and the other two terms are bounded above. If $r[f_j] = 0$ then the first term approaches $q_{\boldsymbol{\lambda}}[f_j]$ and the second term decreases without bound because, by non-degeneracy of U_j, there exists $\varepsilon > 0$ such that $\mathrm{U}_{r,j}(-\varepsilon) = \mathrm{U}_j(\varepsilon) < \infty$ and hence by Fenchel's inequality $-\mathrm{U}_{r,j}^*(\lambda_j + \delta) \le (\lambda_j+\delta)\varepsilon + \mathrm{U}_{r,j}(-\varepsilon)$.

References

Y. Altun and A. Smola. Unifying divergence minimization and statistical inference via convex duality. In *Proceedings of the Nineteenth Annual Conference on Learning Theory*, 2006.

R. P. Anderson and E. Martínez-Meyer. Modeling species' geographic distributions for preliminary conservation assessments: an implementation with the spiny pocket mice (*Heteromys*) of Ecuador. *Biological Conservation*, 116:167–179, 2004.

A. L. Berger, S. A. Della Pietra, and V. J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

S. N. Bernstein. *Theory of Probability*. Gostekhizdad, 1946.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

L. M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *U.S.S.R. Computational Mathematics and Mathematical Physics*, 7(1):200–217, 1967.

N. Cesa-Bianchi, A. Krogh, and M. K. Warmuth. Bounds on approximate steepest descent for likelihood maximization in exponential families. *IEEE Transactions on Information Theory*, 40 (4):1215–1220, July 1994.

S. F. Chen and R. Rosenfeld. A survey of smoothing techniques for ME models. *IEEE Transactions on Speech and Audio Processing*, 8(1):37–50, January 2000.

M. Collins, R. E. Schapire, and Y. Singer. Logistic regression, AdaBoost and Bregman distances. *Machine Learning*, 48(1):253–285, 2002.

J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, 43(5):1470–1480, 1972.

O. Dekel, S. Shalev-Shwartz, and Y. Singer. Smooth ε-insensitive regression by loss symmetrization. In *Proceedings of the Sixteenth Annual Conference on Computational Learning Theory*, pages 433–447. Springer, 2003.

S. Della Pietra, V. Della Pietra, and J. Lafferty. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):1–13, April 1997.

S. Della Pietra, V. Della Pietra, and J. Lafferty. Duality and auxiliary functions for Bregman distances. Technical Report CMU-CS-01-109, School of Computer Science, Carnegie Mellon University, 2001.

L. Devroye. Bounds for the uniform deviation of empirical measures. *Journal of Multivariate Analysis*, 12:72–79, 1982.

D. L. Donoho and M. Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ^1 minimization. *Proceedings of the National Academy of Sciences*, 100(5):2197–2202, March 2003.

D. L. Donoho and I. M. Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81 (3):425–455, August 1994.

M. Dudík and R. E. Schapire. Maximum entropy distribution estimation with generalized regularization. In *Proceedings of the Nineteenth Annual Conference on Learning Theory*, pages 123–138. Springer-Verlag, 2006.

M. Dudík, R. E. Schapire, and S. J. Phillips. Correcting sample selection bias in maximum entropy density estimation. In *Advances in Neural Information Processing Systems 18*, pages 323–330. MIT Press, 2005.

J. Elith. Quantitative methods for modeling species habitat: Comparative performance and an application to Australian plants. In Scott Ferson and Mark Burgman, editors, *Quantitative Methods for Conservation Biology*, pages 39–58. Springer-Verlag, New York, 2002.

J. Elith, C. H. Graham, and the NCEAS Species Distribution Modelling Group. Novel methods improve prediction of species' distributions from occurrence data. *Ecography*, 29(2):129–151, 2006.

S. Ferrier, M. Drielsma, G. Manion, and G. Watson. Extended statistical approaches to modelling spatial pattern in biodiversity: the north-east New South Wales experience. II. Community-level modelling. *Biodiversity and Conservation*, 11:2309–2338, 2002.

J. Goodman. Sequential conditional generalized iterative scaling. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 9–16, July 2002.

J. Goodman. Exponential priors for maximum entropy models. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2004.

C. H. Graham, C. Moritz, and S. E. Williams. Habitat history improves prediction of biodiversity in rainforest fauna. *Proceedings of the National Academy of Sciences of the United States of America*, 103(3):632–636, January 2006.

L. Hannah, G. Midgley, G. Hughes, and B. Bomhard. The view from the Cape: Extinction risk, protected areas, and climate change. *BioScience*, 55(3), March 2005.

A. E. Hoerl and R. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970.

G. E. Hutchinson. Concluding remarks. *Cold Spring Harbor Symposia on Quantitative Biology*, 22:415–427, 1957.

E. T. Jaynes. Information theory and statistical mechanics. *Physics Reviews*, 106:620–630, 1957.

B. M. Jedynak and S. Khudanpur. Maximum likelihood set for estimating a probability mass function. *Neural Computation*, 17:1508–1530, 2005.

J. Kazama and J. Tsujii. Evaluation and extension of maximum entropy models with inequality constraints. In *Conference on Empirical Methods in Natural Language Processing*, pages 137–144, 2003.

S. P. Khudanpur. A method of maximum entropy estimation with relaxed constraints. In *Proceedings of the Johns Hopkins University Language Modeling Workshop*, pages 1–17, 1995.

B. Krishnapuram, L. Carin, M. A. T. Figueiredo, and A. J. Hartemink. Sparse multinomial logistic regression: Fast algorithms and generalization bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):957–968, June 2005.

R. Lau. Adaptive statistical language modeling. Master's thesis, MIT Department of Electrical Engineering and Computer Science, May 1994.

J. R. Leathwick, J. Elith, M. P. Francis, T. Hastie, and P. Taylor. Variation in demersal fish species richness in the oceans surrounding New Zealand: an analysis using boosted regression trees. *Marine Ecology Progress Series*, 321:267–281, 2006.

J. R. Leathwick, D. Rowe, J. Richardson, J. Elith, and T. Hastie. Using multivariate adaptive regression splines to predict the distributions of New Zealand's freshwater diadromous fish. *Freshwater Biology*, 50:2034–2051, 2005.

G. Lebanon and J. Lafferty. Boosting and maximum likelihood for exponential models. Technical Report CMU-CS-01-144, CMU School of Computer Science, October 2001.

R. Malouf. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the Sixth Conference on Natural Language Learning*, pages 49–55, 2002.

C. McDiarmid. On the method of bounded differences. In *Surveys in Combinatorics 1989*, pages 148–188. Cambridge University Press, 1989.

G. G. Moisen and T. S. Frescino. Comparing five modeling techniques for predicting forest characteristics. *Ecological Modeling*, 157:209–225, 2002.

M. New, M. Hulme, and P. Jones. Representing twentieth-century space-time climate variability. Part 1: Development of a 1961-90 mean monthly terrestrial climatology. *Journal of Climate*, 12: 829–856, 1999.

W. I. Newman. Extension to the maximum entropy method. *IEEE Transactions on Information Theory*, IT-23(1):89–93, January 1977.

A. Y. Ng. Feature selection, L_1 vs. L_2 regularization, and rotational invariance. In *Proceedings of the Twenty-First International Conference on Machine Learning*, pages 615–622, 2004.

A. T. Peterson. Predicting species' geographic distributions based on ecological niche modeling. *The Condor*, 103:599–605, 2001.

A. T. Peterson and J. Shaw. *Lutzomyia* vectors for cutaneous leishmaniasis in southern Brazil: Ecological niche models, predicted geographic distribution, and climate change effects. *International Journal of Parasitology*, 33:919–931, 2003.

S. J. Phillips, R. P. Anderson, and R. E. Schapire. Maximum entropy modeling of species geographic distributions. *Ecological Modelling*, 190(3–4):231–259, 2006.

S. J. Phillips, M. Dudík, and R. E. Schapire. A maximum entropy approach to species distribution modeling. In *Proceedings of the Twenty-First International Conference on Machine Learning*, pages 655–662, 2004.

W. F. Ponder, G. A. Carter, P. Flemons, and R. R. Chapman. Evaluation of museum collection data for use in biodiversity assessment. *Conservation Biology*, 15:648–657, 2001.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

R. Rosenfeld. A maximum entropy approach to adaptive statistical language modeling. *Computer, Speech and Language*, 10:187–228, 1996.

S. Rosset and E. Segal. Boosting density estimation. In *Advances in Neural Information Processing Systems 15*, pages 641–648. MIT Press, 2003.

R. Salakhutdinov, S. T. Roweis, and Z. Ghahramani. On the convergence of bound optimization algorithms. In *Uncertainty in Artificial Intelligence 19*, pages 509–516, 2003.

J. R. Sauer, J. E. Hines, and J. Fallon. The North American breeding bird survey, results and analysis 1966–2000, Version 2001.2. http://www.mbr-pwrc.usgs.gov/bbs/bbs.html, 2001. USGS Patuxent Wildlife Research Center, Laurel, MD.

R. E. Schapire. The boosting approach to machine learning: An overview. In *MSRI Workshop on Nonlinear Estimation and Classification*, 2002.

D. R. B. Stockwell and A. T. Peterson. Controlling bias in biodiversity data. In J. Michael Scott, Patricia J. Heglund, Michael L. Morrison, Jonathan B. Haufler, Martin G. Raphael, William A. Wall, and Fred B. Samson, editors, *Predicting Species Occurrences: Issues of Accuracy and Scale*, pages 537–546. Island Press, Washington, DC, 2002.

R. Tibshirani. Regression shrikage and selection via the lasso. *J. R. Statist. Soc. B*, 58(1):267–288, 1996.

USGS. HYDRO 1k, elevation derivative database. Available at http://edcdaac.usgs.gov/gtopo30/hydro/, 2001. United States Geological Survey, Sioux Falls, South Dakota.

E. Welk, K. Schubert, and M. H. Hoffmann. Present and potential distribution of invasive mustard (*Alliara petiolata*) in North America. *Diversity and Distributions*, 8:219–233, 2002.

M. Welling, R. S. Zemel, and G. E. Hinton. Self supervised boosting. In *Advances in Neural Information Processing Systems 15*, pages 665–672. MIT Press, 2003.

P. M. Williams. Bayesian regularization and pruning using a Laplace prior. *Neural Computation*, 7 (1):117–143, 1995. ISSN 0899-7667.

T. Zhang. Class-size independent generalization analysis of some discriminative multi-category classification. In *Advances in Neural Information Processing Systems 17*, pages 1625–1632, 2005.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *J. R. Statist. Soc. B*, 67:301–320, 2005.

Journal of Machine Learning Research 8 (2007) 1261-1276 Submitted 3/06; Revised 10/06; Published 6/07

Measuring Differentiability: Unmasking Pseudonymous Authors

Moshe Koppel KOPPEL@CS.BIU.AC.IL
Jonathan Schler SCHLERJ@CS.BIU.AC.IL
Elisheva Bonchek-Dokow LE7.BONCHEK.DOKOW@GMAIL.COM
Department of Computer Sciences
Bar-Ilan University
Ramat-Gan 52900, Israel

Editor: Michael Collins

Abstract

In the authorship verification problem, we are given examples of the writing of a single author and are asked to determine if given long texts were or were not written by this author. We present a new learning-based method for adducing the "depth of difference" between two example sets and offer evidence that this method solves the authorship verification problem with very high accuracy. The underlying idea is to test the rate of degradation of the accuracy of learned models as the best features are iteratively dropped from the learning process.

Keywords: authorship attribution, one-class learning, unmasking

1 Introduction

In the *authorship attribution* problem, we are given examples of the writing of a number of authors and are asked to determine which of them authored given anonymous texts (Mosteller and Wallace 1964; Holmes 1998). If it can be assumed for each test document that one of the specified authors is indeed the actual author, the problem fits the standard paradigm of a text categorization problem (Sebastiani 2002).

In the *authorship verification* problem (Van Halteren 2004), we are given examples of the writing of a single author and are asked to determine if given texts were or were not written by this author. As a categorization problem, verification is significantly more difficult than attribution and little, if any, work has been performed on it in the learning community. If, for example, all we wished to do is to determine if a text was written by Shakespeare or Marlowe, it would be sufficient to use their respective known writings, to construct a model distinguishing them, and to test the unknown text against the model. If, on the other hand, we need to determine if a text was written by Shakespeare or not, it is very difficult – if not impossible – to assemble an exhaustive, or even representative, sample of not-Shakespeare. The situation in which we suspect that a given author may have written some text but do not have an exhaustive list of alternative candidates is a common one. The problem is complicated by the fact that a single author may consciously vary his or her style from text to text for many reasons or may unconsciously drift stylistically over time. Thus researchers must learn to somehow distinguish between relatively shallow differences that reflect conscious or unconscious changes in an author's style and deeper differences that reflect styles of different authors.

Verification is thus essentially a one-class problem. A fair amount of work has been done on one-class problems, especially using support vector machines (Manevitz and Yousef 2001; Schölkopf et al., 2001; Tax 2001). There are, however, two important points that must be noted. First, in authorship verification we do not actually lack for negative examples. The world is replete with texts that were, for example, not written by Shakespeare. However, these negative examples are not representative of the entire class. Thus, for example, the fact that a particular text may be deemed more similar to the known works of Shakespeare than to those of some given set of other authors does not by any means constitute solid evidence that the text was authored by Shakespeare rather than by some other author not considered at all. We will consider how to make proper limited use of whatever partial negative information is available for the authorship verification problem.

A second distinction between authorship verification and some one-class problems is that if the text we wish to attribute is long – and in this paper we will consider only long texts – then we can chunk the text so that we effectively have multiple examples which are known to be either all written by the author or all not written by the author. Thus, a better way to think about authorship verification is that we are given two example sets and are asked whether these sets were generated by a single generating process (author) or by two different processes.

The central novelty of this paper is a new method for adducing the depth of difference between two example sets, a method that may have far-reaching consequences for determining the reliability of classification models. The underlying idea is to test the rate of degradation of the accuracy of learned models as the best features are iteratively dropped from the learning process.

We will show that this method provides an extremely robust solution to the authorship verification problem that is independent of language, period and genre. This solution has already been used to settle at least one outstanding literary problem.

2 Authorship Attribution with Two Candidates

Since our methods will build upon the standard methods for handling authorship attribution between two candidate authors, let us begin by briefly reviewing those standard methods.

We begin by choosing a feature set consisting of the kinds of features that might be used consistently by a single author over a variety of writings. Typically, these features might include frequencies of function words (Mosteller and Wallace 1964), syntactic structures (Baayen et al., 1996; Stamatatos et al., 2001), parts-of-speech n-grams (Koppel et al., 2002), complexity and richness measures (such as sentence length, word length, type/token ratio) (Yule 1938; Tweedie and Baayen 1998; De Vel et al., 2002) or syntactic and orthographic idiosyncrasies (Koppel and Schler 2003). Note that these feature types contrast sharply with the content words commonly used in text categorization by topic.

Having constructed the appropriate feature vectors, we continue, precisely as in topic-based text categorization, by using a learning algorithm to construct a distinguishing model. Although many learning methods have been applied to the problem, including multivariate analysis, decision trees and neural nets, a good number of studies have shown that linear separators work well for text categorization (Yang 1999). Linear methods that have been used for text categorization include Naïve Bayes (Mosteller and Wallace 1964; Peng et al., 2004), which for the two-class case is a linear separator, Winnow and Exponential Gradient (Lewis et al., 1996; Dagan et al., 1997; Koppel et al., 2002) and linear support vector machines (SVM) (Joachims 1998; De Vel et al., 2002; Diederich et al., 2003).

This general framework has been used to convincingly solve a number of real world authorship attribution problems (e.g., Mosteller and Wallace 1964; Matthews and Merriam 1993; Holmes et al., 2001; Binongo 2003).

3 Authorship Verification: Naïve Approaches

Is there some way to leverage these methods to solve the verification problem in which we are asked if some book X was written by author A without being offered a closed set of alternatives? Let us begin by considering three naïve approaches to the problem. Although none of them will prove satisfactory, each will contribute to our understanding of the problem.

Approach 1: Lining up impostors – One possibility that suggests itself is to assemble a representative collection of works by other authors and to use a two-class learner, such as SVM, to learn a model for A vs. not-A. Then we chunk the mystery work X and run the chunks through the learned model. If the preponderance of chunks of X are classed as A, then X is deemed to have been written by A; otherwise it is deemed to have been written by someone else. This method is straightforward but it suffers from a conceptual flaw. While it is indeed reasonable to conclude that A is not the author if most chunks are attributed to not-A, the converse is not true. Any author who is neither A nor represented in the sample not-A, but who happens to have a style more similar to A than to not-A, will be falsely determined by this method to be A. Despite this flaw, we will see later that this approach can be used to augment other methods.

Approach 2: One-class learning – Another plausible approach would be to try to handle the problem with *no* negative examples at all. The most straightforward way to do this is to apply a one-class learner, such as a one-class support vector machine (Chang and Lin 2001), that finds an optimal boundary that circumscribes all positive examples of A. Then we ascribe X to A if a sufficient number of chunks of X lie inside the boundary. This approach is conceptually sound but we will see that it performs poorly in practice for our problem.

Approach 3: Comparing A directly to X – Another approach that doesn't depend on negative examples is this: learn a model for A vs. X and assess the extent of the difference between A and X using cross-validation. If it is easy to distinguish between them, that is, if we obtain high accuracy in cross-validation, then conclude that A did not write X. If we fail to correctly classify test examples, conclude that A did write X. This method does not work well at all. But since the method that we introduce in this paper is based on this method, it is worth our while to pause here and consider exactly why the method fails.

Let's consider a real-world example. We are given known works by three 19th century American novelists, Herman Melville, James Fenimore Cooper and Nathaniel Hawthorne. For each of the three authors, we are asked if that author was or was not also the author of *The House of Seven Gables* (henceforth: *Gables*). Using the method just described and using a feature set consisting of the 250 most frequently used words in A and X (details below), we find that we can distinguish *Gables* from the works of each author with cross-validation accuracy of above 98%. If we were to conclude, therefore, that none of these authors wrote *Gables*, we would be wrong: Hawthorne wrote it.

4 A New Approach: Unmasking

If we look closely at the models that successfully distinguish *Gables* from Hawthorne's other work (in this case, *The Scarlet Letter*), we find that a small number of features are doing all the work of distinguishing between them. These features include *he* (more frequent in *The Scarlet Letter*) and *she* (more frequent in *Gables*). The situation in which an author will use a small number of features in a consistently different way between works is typical. These differences might result from thematic differences between the works, from differences in genre or purpose, from chronological stylistic drift, or from deliberate attempts by the author to mask his or her identity (as we shall see below).

The main point of this paper is to show how this problem can be overcome by determining not only if A is distinguishable from X but also how great is the depth of difference between A and X. To do this we introduce a new technique we call "unmasking". The intuitive idea of unmasking is to iteratively remove those features that are most useful for distinguishing between A and X and to gauge the speed with which cross-validation accuracy degrades as more features are removed. Our main hypothesis is that if A and X are by the same author, then whatever differences there are between them will be reflected in only a relatively small number of features, despite possible differences in theme, genre and the like.

In Figure 1, we show the result of unmasking when comparing *Gables* to known works of Melville, Cooper and Hawthorne. This graph illustrates our hypothesis: when comparing *Gables* to works by other authors the degradation as we remove distinguishing features from consideration is slow and smooth but when comparing it to another work by Hawthorne, the degradation is sudden and dramatic. This illustrates that once a small number of distinguishing markers are removed, the two works by Hawthorne become increasingly hard to distinguish from each other.

In the following section, we will show how the suddenness of the degradation can be quantified in a fashion optimal for this task and we will see that the phenomenon illustrated in the *Gables* example holds very generally. Thus by taking into account the depth of difference between two works, we can determine if they were authored by the same person or two different people.

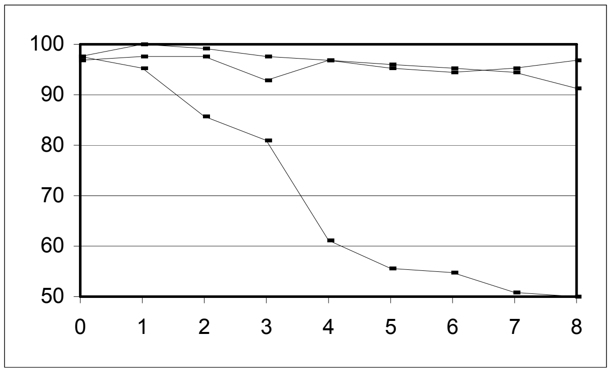


Figure 1. Ten-fold cross-validation accuracy of models distinguishing *House of Seven Gables* from each of Hawthorne, Melville and Cooper. The *x*-axis represents the number of iterations of eliminating best features at previous iteration. The curve well below the others is that of Hawthorne, the actual author.

5 Experimental Results

In this section we consider an experimental framework for systematic testing of the effectiveness of the unmasking algorithm

5.1 Corpus

We consider a collection of 21 nineteenth century English books written by 10 different authors and spanning a variety of genres. We chose all electronically available books by these authors that were sufficiently large (above 500K) and that presented no special formatting challenges. The full list is shown in Table 1. Our objective is to run 209 independent authorship verification experiments representing all possible author/book pairs (21 books × 10 authors but excluding the pair Emily Bronte/*Wuthering Heights* which can't be tested since it is the author's only work).

Group	Author	Book	Chunks
American Novelists	Hawthorne	Dr. Grimshawe's Secret	75
		House of Seven Gables	63
	Melville	Redburn	51
		Moby Dick	88
	Cooper	The Last of the Mohicans	49
		The Spy	63
		Water Witch	80
American Essayists	Thoreau	Walden	49
		A Week on Concord	50
	Emerson	Conduct Of Life	47
		English Traits	52
British Playwrights	Shaw	Pygmalion	44
		Misalliance	43
		Getting Married	51
	Wilde	An Ideal Husband	51
		Woman of No Importance	38
Bronte Sisters	Anne	Agnes Grey	45
		Tenant Of Wildfell Hall	84
	Charlotte	The Professor	51
		Jane Eyre	84
	Emily	Wuthering Heights	65

Table 1 The list of books used in our experiments.

For the sake of all the experiments that follow, we chunk each book into approximately equal sections of at least 500 words without breaking up paragraphs. For each author A and each book X, let A_X consist of all the works by A in the corpus unless X is in fact written by A, in which case A_X consists of all works by A except X. Our objective is to assign to each pair $<A_X,X>$ the value *same-author* if X is by A and the value *different-author* otherwise.

5.2 Baseline: One-class SVM

In order to establish a baseline, for each author A in the corpus and each book X, we use a one-class SVM (Chang and Lin 2001) on the 250 most frequent words in A_X to build a model for A_X.

(Although, as discussed above, many more interesting feature sets are possible, we use this feature set here for simplicity and universal applicability.) We then test each book X against the model of each A_X. We assign the pair $<A_X,X>$ the value *same-author* if more than half the chunks of X are assigned to A_X. This method performs very poorly. Of the 20 pairs that should have been assigned the value *same-author*, only 6 are correctly classified, while 46 of the 189 pairs that should be assigned the value *different-author* are incorrectly classified. These results hold using an RBF kernel; using other kernels or using a threshold other than half (the number of chunks assigned to the class) only degrades results.

A second possible baseline is the "lining up impostors" method mentioned in Section 3 above. We will discuss this method in some detail in Section 6.

5.3 Unmasking Applied

Now let us introduce the details of our new method based on our observations above regarding iterative elimination of features. We choose as an initial feature set the *n* words with highest average frequency in A_X and X (that is, the average of the frequency in A_X and the frequency in X, giving equal weight to A_X and X). Using an SVM with linear kernel we run the following unmasking scheme:

1. Determine the accuracy results of a ten-fold cross-validation experiment for A_X against X. (If one of the sets, A_X or X, includes more chunks than the other, we randomly discard the surplus. Accuracy results are the average of five runs of ten-fold cross-validation in which we discard randomly for each run.)
2. For the model obtained in each fold, eliminate the *k* most strongly weighted positive features and the k most strongly weighted negative features.
3. Go to step 1.

In this way, we construct degradation curves for each pair $<A_X,X>$. In Figure 2, we show such curves (using *n*=250 and *k*=3) for *An Ideal Husband* against each of ten authors, including Oscar Wilde.

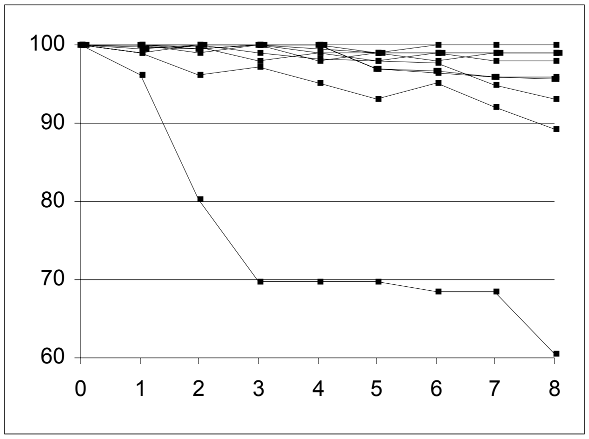


Figure 2. Unmasking *An Ideal Husband* against each of the ten authors (n=250, k=3). The curve below all the authors is that of Oscar Wilde, the actual author. (Several curves are indistinguishable.)

5.4 Meta-learning: Identifying *Same-Author* Curves

We wish now to quantify the difference between *same-author* curves and *different-author* curves. To do so, we first represent each curve as a numerical vector in terms of its essential features. These features include, for $i = 0,\dots,m$:

- accuracy after i elimination rounds
- accuracy difference between round i and i+1
- accuracy difference between round i and i+2
- i^{th} highest accuracy drop in one iteration
- i^{th} highest accuracy drop in two iterations

We sort these vectors into two subsets: those in which A_X and X are the by same author and those in which A_X and X are by different authors. We then apply a meta-learning scheme in which we use learners to determine what role to assign to various features of the curves. (Note that although we have 20 *same-author* pairs, we really only have 13 distinct *same-author* curves, since for authors with exactly two works in our corpus, the comparison of A_X with X is identical for each of the two books.)

To illustrate the ease with which *same-author* curves can be distinguished from *different-author* curves, we note that for all *same-author* curves, it holds that:

- accuracy after 6 elimination rounds is lower than 89% *and*
- the second highest accuracy drop in two iterations is greater than 16%.

These two conditions hold for only 5 of the 189 *different-author* curves.

5.5 Accuracy Results: Leave-one-book-out Tests

In order to assess the accuracy of the method, we use the following cross-validation methodology. For each book B in our corpus, we run a trial in which B is completely eliminated from consideration. We use unmasking to construct curves for all author/book pairs $<A_X,X>$ (where B does not appear in A_X and is not X) and then we use a linear SVM to meta-learn to distinguish *same-author* curves from *different-author* curves. Then, for each author A in the corpus, we use unmasking to construct a curve for the pair $<A_B,B>$ and use the meta-learned model to determine if the curve is a *same-author* curve or a *different-author* curve.

Using this testing protocol, we obtain the following results: All but one of the twenty *same-author* pairs are correctly classified. The single exception is *Pygmalion* by George Bernard Shaw. In addition, 181 of 189 *different-author* pairs were correctly classified. Among the exceptions were the attributions of *The Professor* by Charlotte Bronte to each of her sisters. Thus, we obtain overall accuracy of 95.7% with errors almost identically distributed between false positives and false negatives. (It should be noted that some of the 8 misclassified *different-author* pairs result in a single book being attributed to two authors, which is obviously impossible. Nevertheless, since each of our author/book pairs is regarded as an independent experiment, we do not leverage this information.)

Note that the algorithm includes three parameters: *n*, the size of the initial feature set; *k*, the number of eliminated features from each extreme in each iteration; *m*, the number of iterations we consider. The results reported above are based on experiments using *n*=250, *k*=3, and *m*=10, the settings used in initial experiments first reported in Koppel and Schler (2004). We chose n=250 because experimentation indicated that this was a reasonable rough boundary between common words and words tightly tied to a particular work. Nevertheless, it might be asked how robust our results are vis-à-vis these parameters. To test this, we ran our leave-one-book-out experiment for a variety of parameter settings. The results are shown in Table 2.

Features (n)	Features eliminated (k)	Iterations (m)	Correctly classified *same-author* (out of 20)	Correctly classified *different-author* (out of 189)	F1 (*macro average*)
250	3	5	16	183	0.868
		10	19	181	0.892
		20	20	180	0.896
	6	5	20	182	0.916
		10	20	180	0.896
		20	20	181	0.906
	10	5	20	180	0.896
		10	20	179	0.886
500	3	5	14	189	0.904
		10	12	186	0.828
		20	16	180	0.838
	6	5	13	184	0.826
		10	18	180	0.868
		20	19	179	0.873
	10	5	16	181	0.848
		10	18	180	0.868
		20	20	177	0.868
1000	3	5	11	189	0.843
		10	11	188	0.831
		20	12	183	0.797
	6	5	12	188	0.852
		10	14	184	0.844
		20	17	181	0.863
	10	5	15	184	0.862
		10	16	182	0.857
		20	16	177	0.812

Table 2 Accuracy results on the 21 book experiment for a variety of parameter setting

As is evident, results are somewhat robust with regard to choice of *k* and *m* (in fact, some parameter choices turn out to be better than our initial choice), but the recall results for *same-author* degrades considerably as the size of the initial feature set increases. Apparently, what is important is that at some stage a sufficiently small feature set is reached. A related pattern that is evident in the data is that as the maximum number of features eliminated (i.e., k*m) increases, the total number of example pairs classified as *same-author* increases. (This is reflected by increasing *same-author* recall and decreasing *different-author* recall.) This is because it is the behavior of significantly stripped-down feature sets that permits meta-learning to effectively distinguish between *same-author* curves and *different-author* curves. In the absence of such information, support vector machines tend to err in the direction of majority class, which in this case is *different-author*.

6 Extension: Using Negative Examples

Until now we have not used any examples of non-A writing to help us construct a model of author A. Of course, plenty of examples of non-A writing exist; they are simply neither exhaustive nor representative. We now consider how use can be made of such data.

Let us recall the "lining up impostors" method suggested in Section 3 above. Suppose, that we have available the works of several authors roughly filling the same profile as A in terms of geography, chronology, culture and genre. To make matters concrete, suppose we are considering whether some book X was written by Melville. We could use the works of Hawthorne and Cooper as examples of non-Melville writing and learn a model for Melville against non-Melville. Assuming we can do so successfully, we can then test each example of X to see if it assigned by the model to Melville or to not-Melville. If many are assigned to not-A, it might be reasonable to conclude that X is not by the same author as A. But, as we noted above, if it turns out that many, or even all, examples of X are assigned to Melville, we would be hard-pressed to conclude that Melville wrote the book since it may very well be that the works of other authors, say Shaw or Bronte, happen to be more similar to Melville than to Hawthorne or Cooper.

Nevertheless, it is instructive to try this method. Formally, we proceed as follows. For each author A, choose other authors of the same type ("impostors") – let's call them A1,…,An – and allow them to collectively represent the class not-A. In our corpus, we consider four types as indicated in Table 1. We learn a model for A against not-A and we learn models for each Ai against not-Ai. Assuming that k-fold cross-validation results for each of these models are satisfactory, we test all the examples in X against each one of these models. Let A(X) be the percentage of examples of X classed as A rather than not A; define Ai(X) analogously. Then if A(X) is not greater than Ai(X) for all i, conclude that A is not the author of X. Otherwise conclude that A is the author of X.

Applying this impostors method to our 209 book-author experiments, we find that 19 of 20 *same-author* pairs are correctly classified but only125 of 189 *different-author* pairs are correctly classified. If we try to remedy the tendency of the method to over-assign to the class *same-author* by adding the constraint that X not be assigned to A unless A(X) exceeds some threshold, θ, we at best obtain only very slightly improved performance. For example, for θ=1/2, 19 of 20 *same-author* pairs and 127 of 189 *different-author* pairs are correctly classified. For θ=.8, 13 of 20 *same-author* pairs and 140 of 189 *different-author* pairs are correctly classified.

In short, the basic impostors method often wrongly concludes that a given author wrote a given book, but when it concludes that a given author did *not* write a given book (because some impostor looks more plausible), it is almost always correct. Thus, although the impostors method is obviously not as effective as unmasking as a stand-alone method, it can be used to augment unmasking. We simply conclude that A is the author of X if and only if both the unmasking method and the impostors method indicate that A is the author of X. If either of them indicates that A is not the author of X, we conclude that A is not the author of X.

```
Given: anonymous book X, works of suspect author A,
       (optionally)impostors {A1,…,An}

Step 1 – Impostors method(optional)

if impostors {A1,…,An} are given then
{
    Build model M for classifying A vs. all impostors
    Test each chunk of X with built model M
    foreach impostor Ai
    {
         Build model Mi for classifying Ai vs. {A ∪ all other
                                                  impostors}
         Test each chunk of X with built model Mi
     }
     If for some Ai number of chunks assigned to Ai > number of
                                     chunks assigned to A
     then
           return different-author
}

Step 2 - Unmasking
Build degradation curve <A,X>
Represent degradation curve as feature vector (see text)
Test degradation curve vector (see text)
    if test result positive
        return same-author
    else
        return different-author

Method Build Degradation Curve:

Use 10 fold cross validation for A against X
foreach fold
{
      Do m iterations
      {
            Build a model for A against X
            Evaluate accuracy results
            Add accuracy number to degradation curve <A,X>
            Remove k top contributing features (in each
                                 direction) from data
       }
}
```
Figure 3: Overview of the authorship verification algorithm.

In our experiment, using the augmenting unmasking with the impostors method resulted in the introduction of a single new misclassification: Thoreau was incorrectly concluded not to have written *A Week on Concord.* At the same time, all of the *different-author* pairs previously misclassified as *same-author* were corrected. Overall, then, the augmented method classed all 189 *different-author* pairs and 18 of 20 *same-author* pairs correctly.

In Figure 3, we summarize the entire algorithm including both unmasking and (optionally) the impostors method. Note that although we introduced the impostors method after the unmasking method in our exposition, for purposes of the pseudo-code it is more natural to present the impostors method as a filter prior to the unmasking method.

7 Effects of Topic Variability on Unmasking

It might well be wondered how robust unmasking is to variability in topic. Specifically, can we successfully identify two works as being by a single author even if they are on different topics and, conversely, can we identify two works as being by two different authors even if they are on the same topic?

An ideal corpus for testing this question is that of rabbinic legal responsa. This corpus of Hebrew-Aramaic letters in response to legal queries is divided by author and typically sub-divided by general topic: ritual law, family law and business law. For our purposes, we chose three prolific authors all of whom worked in the second half of the 20th century. Table 3 shows the number of responsa written by each author in each of three different topic areas.

	Ritual	Business	Family
Author 1 (Yosef)	328	55	143
Author 2 (Feinstein)	157	46	120
Author 3 (Halberstam)	138	70	82

Table 3. Number of responsa written by each author on each topic in legal responsa corpus.

Using the same parameter settings as above (n=250, k=3, m=10), we plotted a variety of unmasking curves. In Figure 4, we show unmasking curves for each author vs. all other authors (solid lines) where all writings are on the same topic, and unmasking curves for each author's writing on a given topic vs. that same author's writing on all other topics (dotted lines).

As is evident, for *different-author* pairs (on a single topic), accuracy remains high even as features are eliminated, while for *same-author* pairs (on different topics), accuracy degrades as features are eliminated. Evidently, among common words, the set of markers of authorial style, regardless of topic, is much larger than the quickly eliminated set of topic markers. In this sense, it is much easier to tell apart one author from another – even when the authors are writing on the same topic – than to tell apart works on different topics written by the same author. As a result, *same-author* curves and *different-author* curves do not resemble each other, despite the potentially confounding effects of topic. In fact, in this case *different-author* curves are distinguishable from *same-author* curves already at the first iteration, before any unmasking is performed. But note that as unmasking is performed, the differences become most clear at the sixth iteration just as was the case for the English literature curves.

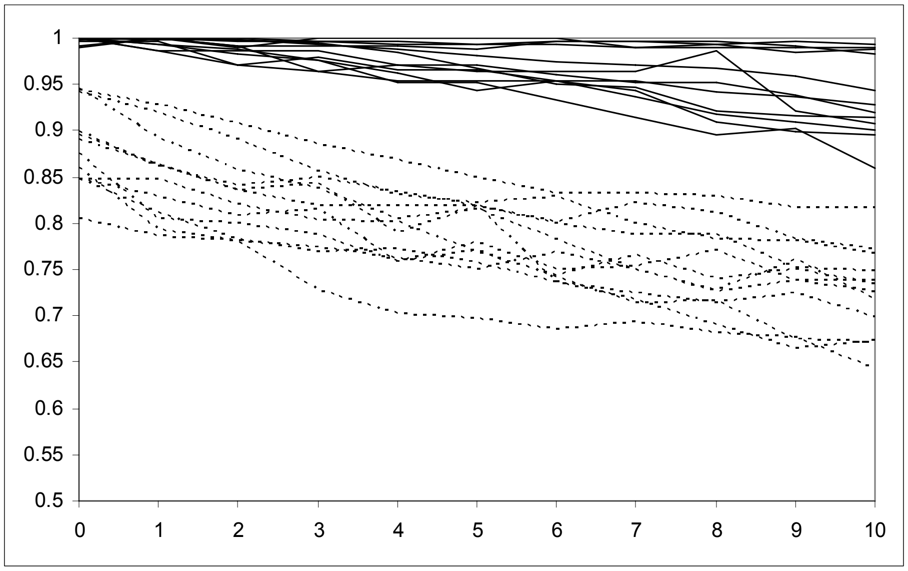


Figure 4 Unmasking of rabbinic legal responsa. Solid lines are *different-author* curves (on same topic) and dotted lines are *same-author* curves (on different topics).

7.1 Solution to a Literary Mystery: The Case of the Bashful Rabbi

Finally, we apply our method to an actual literary mystery. Ben Ish Chai was the leading rabbinic scholar in Baghdad in the late 19th century. Among his vast literary legacy are two collections of responsa. The first, *RP (Rav Pe'alim)* includes 509 documents known to have been authored by Ben Ish Chai. The second, *TL (Torah Lishmah)* includes 524 documents that Ben Ish Chai claims to have found in an archive. There is ample historical reason to believe that he in fact authored the manuscript but did not wish to take credit for it for personal reasons (Ben-David, 2002).

For the sake of comparison, we also have four more collections of responsa written by four other authors working in the same area during the same period. While these far from exhaust the range of possible authors, they collectively constitute a reasonable starting point. There is no reason to believe that any of these authors wrote *TL*.

In any event, the impostors method handily eliminates all candidates but Ben Ish Chai. We now wish to use unmasking to check if Ben Ish Chai is indeed the author. Unmasking is particularly pertinent here, since Ben Ish Chai did not wish to be identified as the author and there is evidence that he may have deliberately altered his style to disguise his authorship. (In fact, the strongest distinguishing features – and hence the first eliminated by unmasking – result from Ben Ish Chai employing different standard signoffs in *RP* and *TL*; it is hard to know whether this reflects deliberate subterfuge or mere chronological drift.)

In Figure 5, we show the results of unmasking for *TL* against Ben Ish Chai as well as, for comparison, each of the other four candidate authors. The curve for Ben Ish Chai is the one far below those of the others. This affirms the consensus among historians (Ben-David 2002) that Ben Ish Chai was indeed the author of TL. Indeed, as in our previous experiments, the differences between the curves are most clear at the sixth iteration.

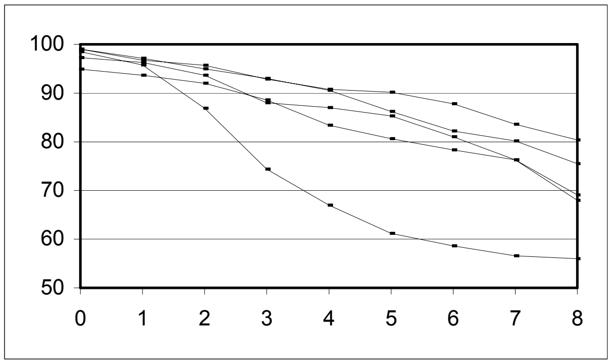


Figure 5: Unmasking *TL* against Ben Ish Chai and four impostors. The curve below the others is Ben Ish Chai.

8 An Alternative Measure of Depth of Difference

It seems plausible that there should be a direct way to determine how different two works are without actually going to the trouble of running the multiple learning experiments required for plotting unmasking curves. One sensible suggestion would be to check the number of features with significant information-gain between authors. Gabrilovich and Markovitch (2004) have used precisely this measure to suggest which learning algorithms might be most appropriate for a given problem.

In Figure 6, we plot the curve in which the *y*-axis represents the number of features with information gain greater than *x*, plotted for each multiple of 0.01 from 0 to 0.6. The ten curves represent the book *An Ideal Husband* vs. the works of each of the ten authors considered in Section 5 above. (Thus, this figure is analogous to Figure 2 above.) The idea is that we expect the curve representing *An Ideal Husband* vs. the other work of Oscar Wilde – the actual author – to show a more sudden drop than those curves comparing *An Ideal Husband* to other authors. And indeed this is the case.

The differences between the *same-author* curve and the *different-author* curves in Figure 6 are not quite as dramatic as the difference in Figure 2 where unmasking curves were used. Nevertheless, the question arises if we can use these curves in much the same way as we use unmasking curves for determining whether two books are by the same author. In order to answer this question, we ran experiments analogous to the leave-one-book-out experiments of Section 5 but with a different feature set. Whereas above we used characteristics of unmasking curves as features in a meta-learning scheme, here we use features of the information-gain curves just considered. More precisely, we record each value shown in Figure 6.

There is no doubt that these curves encode a great deal of information regarding authorship. For example, of the 210 curves generated by the authorship experiment of Section 5, there are 182 IG curves in which there are fewer than 65 features with information gain above 0.03. Of these, 179 are *different-author* curves and only 3 are *same-author* curves. Unfortunately, however, using the meta-learning approach described in Section 5 and a leave-one-out testing protocol results in correct classification of 182 out of 189 *different-author* curves, but only 11 out of 20 *same-author* curves. These results are not quite as good as those obtained using unmasking but they do suggest that information-gain curves are somewhat useful despite their simplicity.

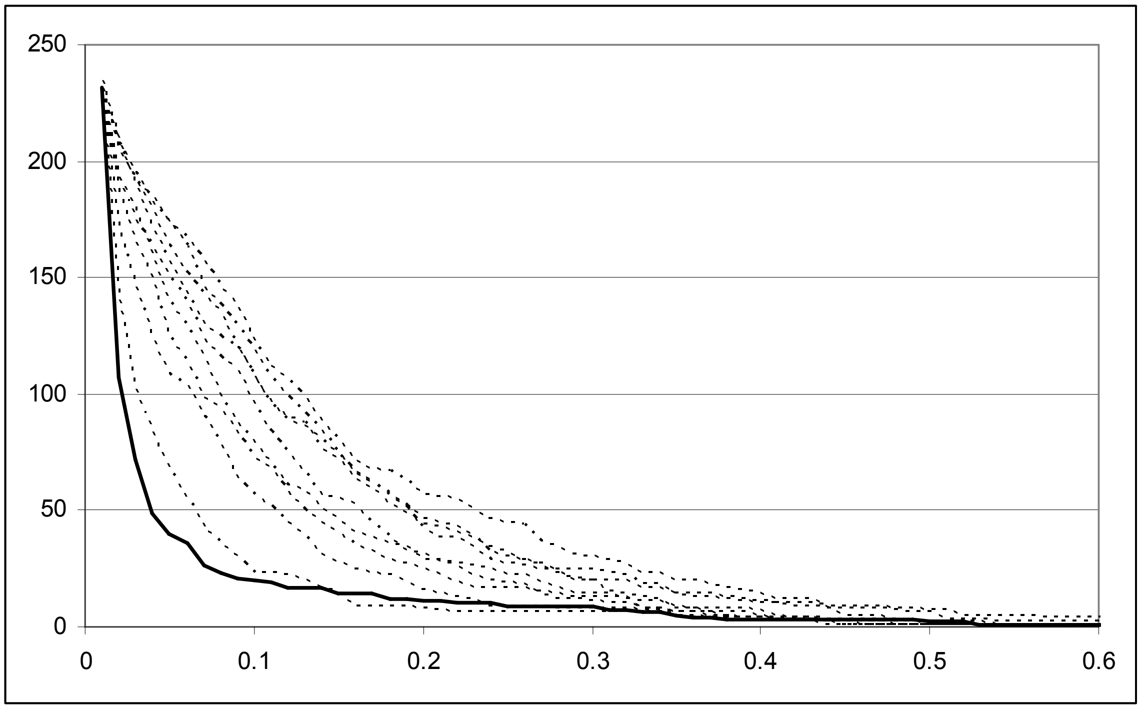


Figure 6. Information-gain curves for *An Ideal Husband* versus ten authors. The dark line is Oscar Wilde, the actual author.

9 Conclusions

The essentials of two-class text categorization are fairly well understood. We have shown in this paper that by using ensembles of text-categorization results as raw material for meta-level analysis, we are able to solve a more difficult and sophisticated problem such as authorship verification. Even when we completely ignore negative examples and thus treat authorship verification as a true one-class classification problem, our methods obtain extremely high accuracy on out-of-sample author/book pairs. When we use just a bit of non-representative negative data, classification is even better.

Nothing in our method is tied to any particular language, period or genre and some evidence presented suggests that similar results are obtained as these parameters are varied. In fact, some evidence presented suggests that the method is immune to deliberate attempts to cover up authorship.

The point of the unmasking method suggested here is to measure of the true "depth of difference" between two example sets. This measure is clearly of a different type than other measures, such as margin width, that could in principle depend on a single highly differentiating feature. Although we have tested the method on a single application, it is not unreasonable to speculate that the new measure presented here ought to be applicable to other applications in which we need to determine whether given phenomena were generated by a single process.

References

H. Baayen, H. Van Halteren and F. Tweedie (1996). Outside the cave of shadows: Using syntactic annotation to enhance authorship attribution, *Literary and Linguistic Computing*, 11, 1996.

Y.L. Ben-David, (2002), Shevet mi-Yehudah (in Hebrew), Jerusalem (no publisher listed)

J.N.G. Binongo, (2003). Who wrote the 15th Book of Oz? An application of multivariate analysis to authorship attribution. *Chance* 16(2), 9-17.

C.C. Chang and C. Lin, (2001) LIBSVM: a Library for Support Vector Machines (Version 2.3)

I. Dagan, Y. Karov and D. Roth (1997), Mistake-driven learning in text categorization, in *EMNLP-97: 2nd Conf. on Empirical Methods in Natural Language Processing*, 1997, pages 55-63.

O. De Vel, M. Corney, A. Anderson and G. Mohay (2002), E-mail authorship attribution for computer forensics, in *Applications of Data Mining in Computer Security*, Barbará, D. and Jajodia, S. (eds.), Kluwer.

J. Diederich, J. Kindermann, E. Leopold and G. Paass (2003), Authorship attribution with support vector machines, *Applied Intelligence 19(1)*, 109-123

E. Gabrilovich and S. Markovitch (2004), Text categorization with many redundant features: using aggressive feature selection to make SVMs competitive with C4.5. In *Proc. 21st International Conference on Machine Learning (ICML) 2004,* pages. 321-328.

D. Holmes (1998). The evolution of stylometry in humanities scholarship, *Literary and Linguistic Computing*, 13, 3, 1998, 111-117.

D. Holmes, L. Gordon, and C. Wilson (2001), A widow and her soldier: Stylometry and the American civil war, *Literary and Linguistic Computing 16(4)*, 403-420

T. Joachims, (1998) Text categorization with support vector machines: learning with many relevant features. In *Proc. 10th European Conference on Machine Learning ECML-98*, pages 137-142

M. Koppel, S. Argamon and A. Shimoni (2002), Automatically categorizing written texts by author gender, *Literary and Linguistic Computing 17(4)*, 401-412

M. Koppel and J. Schler (2003), Exploiting stylistic idiosyncrasies for authorship attribution, in *Proceedings of IJCAI'03 Workshop on Computational Approaches to Style Analysis and Synthesis*, pages 69-72.

M. Koppel and J. Schler (2004), Authorship verification as a one-class classification problem, in *Proceedings of 21st International Conference on Machine Learning*, July 2004, Banff, Canada, pages 489-495.

D. Lewis, D, R. Schapire, J. Callan and R. Papka (1996). Training algorithms for text classifiers, in *Proc. 19th ACM/SIGIR Conf. on R&D in IR*, 1996, pages 306-298.

L Manevitz and M. Yousef (2001). One-class svms for document classification,. *Journal of Machine Learning Research* 2.

R. Matthews and T. Merriam, (1993). Neural computation in stylometry : An application to the works of Shakespeare and Fletcher. *Literary and Linguistic Computing*, 8(4), 203-209.

F. Mosteller and D. L. Wallace (1964). *Inference and Disputed Authorship: The Federalist*. Reading, Mass. : Addison Wesley.

F. Peng, D. Schuurmans and S. Wang (2004). Augmenting naive Bayes text classifier with statistical language models , *Information Retrieval,* 7 (3-4), 317 - 345

B. Schölkopf, J. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson (2001). Estimating the support of a high-dimensional distribution. *Neural Computation*, 13, 1443-1471.

F. Sebastiani, (2002). Machine learning in automated text categorization, *ACM Computing Surveys* 34(1), 1-47.

E. Stamatatos, N. Fakotakis, and G. Kokkinakis (2001) Computer-based authorship attribution without lexical measures. *Computers and the Humanities,* 35(2), 93-214, Kluwer, 2001.

D.M.J. Tax (2001), One-Class Classification. *PhD thesis,* TU Delft, 2001.

F. J Tweedie and R. H. Baayen (1998). How variable may a constant be? Measures of lexical richness in perspective, *Computers and the Humanities,* 32 (1998), 323-352.

H. Van Halteren (2004) Linguistic profiling for authorship recognition and verification, *Proc. of 42nd Conf. Of ACL*, July 2004, 199-206

Y. Yang (1999). An evaluation of statistical approaches to text categorization. *Journal of Information Retrieval*, 1 (1-2), 67-88.

G.U. Yule (1938). On sentence length as a statistical characteristic of style in prose with application to two cases of disputed authorship, *Biometrika*, 30, 363-390.

Journal of Machine Learning Research 8 (2007) 1277-1305 Submitted 6/06; Revised 1/07; Published 6/07

Bayesian Quadratic Discriminant Analysis

Santosh Srivastava SANTOSH@AMATH.WASHINGTON.EDU
Department of Applied Mathematics
University of Washington
Seattle, WA 98195, USA

Maya R. Gupta GUPTA@EE.WASHINGTON.EDU
Department of Electrical Engineering
University of Washington
Seattle, WA 98195, USA

Béla A. Frigyik BFRIGYIK@MATH.PURDUE.EDU
Department of Mathematics
Purdue University
West Lafayette, IN 47907, USA

Editor: Saharon Rosset

Abstract

Quadratic discriminant analysis is a common tool for classification, but estimation of the Gaussian parameters can be ill-posed. This paper contains theoretical and algorithmic contributions to Bayesian estimation for quadratic discriminant analysis. A distribution-based Bayesian classifier is derived using information geometry. Using a calculus of variations approach to define a functional Bregman divergence for distributions, it is shown that the Bayesian distribution-based classifier that minimizes the expected Bregman divergence of each class conditional distribution also minimizes the expected misclassification cost. A series approximation is used to relate regularized discriminant analysis to Bayesian discriminant analysis. A new Bayesian quadratic discriminant analysis classifier is proposed where the prior is defined using a coarse estimate of the covariance based on the training data; this classifier is termed BDA7. Results on benchmark data sets and simulations show that BDA7 performance is competitive with, and in some cases significantly better than, regularized quadratic discriminant analysis and the cross-validated Bayesian quadratic discriminant analysis classifier Quadratic Bayes.

Keywords: quadratic discriminant analysis, regularized quadratic discriminant analysis, Bregman divergence, data-dependent prior, eigenvalue decomposition, Wishart, functional analysis

1. Introduction

A standard approach to supervised classification problems is quadratic discriminant analysis (QDA), which models the likelihood of each class as a Gaussian distribution, then uses the posterior distributions to estimate the class for a given test point (Hastie et al., 2001). The Gaussian parameters for each class can be estimated from training points with maximum likelihood (ML) estimation. The simple Gaussian model assumption is best suited to cases when one does not have much information to characterize a class, for example, if there are too few training samples to infer much about the class distributions. Unfortunately, when the number of training samples n is small compared to the number of dimensions of each training sample d, the ML covariance estimation can be ill-posed.

One approach to resolve the ill-posed estimation is to regularize the covariance estimation; another approach is to use Bayesian estimation.

Bayesian estimation for QDA was first proposed by Geisser (1964), but this approach has not become popular, even though it minimizes the expected misclassification cost. Ripley (2001) in his text on pattern recognition states that such predictive classifiers are mostly unmentioned in other texts and that "this may well be because it usually makes little difference with the tightly constrained parametric families." Geisser (1993) examines Bayesian QDA in detail, but does not show that in practice it can yield better performance than regularized QDA. In preliminary experiments we found that the performance of Bayesian QDA classifiers is very sensitive to the choice of prior (Srivastava and Gupta, 2006), and that priors suggested by Geisser (1964) and Keehn (1965) produce error rates similar to those yielded by ML.

In this paper we propose a Bayesian QDA classifier termed *BDA7*. BDA7 is competitive with regularized QDA, and in fact performs better than regularized QDA in many of the experiments with real data sets. BDA7 differs from previous Bayesian QDA methods in that the prior is selected by crossvalidation from a set of data-dependent priors. Each data-dependent prior captures some coarse information from the training data. Using ten benchmark data sets and ten simulations, the performance of BDA7 is compared to that of Friedman's regularized quadratic discriminant analysis (RDA) (Friedman, 1989), to a model-selection discriminant analysis (EDDA) (Bensmail and Celeux, 1996), to a modern cross-validated Bayesian QDA (QB) (Brown et al., 1999), and to ML-estimated QDA, LDA, and the nearest-means classifier. Our focus is on cases in which the number of dimensions d is large compared to the number of training samples n. The results show that BDA7 performs slightly better than the other approaches averaged over the real data sets. The simulations help analyze the methods under controlled conditions.

This paper also contributes to the theory of Bayesian QDA in several aspects. First, the Bayesian classifier is solved for in terms of the Gaussian distributions themselves, as opposed to the standard approach of formulating the problem in terms of the Gaussian parameters. This distribution-based Bayesian discriminant analysis removes the parameter-based Bayesian analysis restriction of requiring more training samples than feature dimensions, and removes the question of invariance to transformations of the parameters, because the estimate is defined in terms of the Gaussian distribution itself. We show that the distribution-based Bayesian discriminant analysis classifier has roughly the same closed-form as the parameter-based Bayesian discriminant analysis classifier, but has a different degree of freedom.

The second theoretical result links Bayesian QDA and Friedman's regularized QDA by a using series approximation. Third, we show that the Bayesian distribution-based classifier that minimizes the expected misclassification cost is equivalent to the classifier that uses the Bayesian minimum expected Bregman divergence estimates of the class conditional distributions.

We begin with a review of approaches to Bayesian QDA and regularized QDA in Section 2. We formulate the distribution-based Bayesian classifier in Section 3, and review recent results showing that the distribution-based performance is superior to the parameter-based Bayesian classifier given the same prior if no cross-validation is allowed. An approximate relationship between Bayesian QDA and Friedman's regularized QDA is given in Section 4. In Section 5 we establish that the Bayesian minimum expected misclassification cost estimate is equivalent to a plug-in estimate using the Bayesian minimum expected Bregman divergence estimate for each class conditional. Then, we turn to the practical matter of classification: we propose the cross-validated Bayesian QDA classifier BDA7 in Section 6. In Section 7, benchmark data set results compare BDA7 to other QDA

Notation	Description	Notation	Description
I	identity matrix	$I_{(\cdot)}$	indicator function
$x_i \in \mathbb{R}^d$	i^{th} training sample	n	number of training samples
y_i	class label corresponding to x_i	n_h	number of training samples of class h
G	number of class labels	$\bar{x}_h$	sample mean for class h
$\mathcal{T}$	n pairs of training samples	$\mathcal{T}_h$	n_h pairs of class h training samples
$x \in \mathbb{R}^d$	test sample	S	$\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T$
Y	class label corresponding to x	S_h	$\sum_{i=1}^{n}(x_i - \bar{x}_h)(x_i - \bar{x}_h)^T I_{(y_i=h)}$
C	misclassification cost matrix	$\lvert B \rvert$	determinant of B
diag(B)	diagonal of B	tr(B)	trace of B
N	random Gaussian distribution	$\mathcal{N}$	realization of a Gaussian distribution

Table 1: Key Notation

classifiers, followed by further analysis using simulation results in Section 8. The paper concludes with a discussion of the results and some open questions.

Table 1 details some of the notation used in this paper.

2. Prior Research on Ill-Posed QDA

We review the prior literature on Bayesian approaches to QDA, regularization approaches to QDA, and other approaches to resolving ill-posed QDA.

2.1 Bayesian Approaches to QDA

Discriminant analysis using Bayesian estimation was first proposed by Geisser (1964) and Keehn (1965). Geisser's work used a noninformative prior distribution to calculate the posterior odds that a test sample belongs to a particular class. Keehn's work assumed that the prior distribution of the covariance matrix is an inverse Wishart distribution. Raudys and Jain (1991) stated that Bayesian QDA does not solve the problems that occur with ML QDA, particularly when class sample sizes differ. Recent work by the authors showed that Bayesian QDA using the priors suggested by Geisser and Keehn perform similarly to ML QDA on six standard QDA simulations (Srivastava and Gupta, 2006).

The inverse Wishart prior is a conjugate prior for the covariance matrix, and it requires the specification of a "seed" positive definite matrix and a scalar degree of freedom. Following Brown et al. (1999), the term *Quadratic Bayes* (QB) is used to refer to a modern form of Bayesian QDA where the inverse Wishart seed matrix is kI, where I is the d-dimensional identity matrix, k is a scalar, and the parameters k and the degree of freedom q of the inverse Wishart distribution are chosen by crossvalidation.

Previous Bayesian QDA work is *parameter-based* in that the mean μ and covariance Σ are treated as random variables, and the expectations of the Gaussian class-conditional distributions are calculated with respect to Lebesgue measure over the domain of the parameters. In this paper we solve for the *distribution-based* Bayesian QDA, such that the uncertainty is considered to be over the set of Gaussian distributions and the Bayesian estimation is formulated over the domain of the Gaussian distributions.

2.2 Regularizing QDA

Friedman (1989) proposed regularizing ML covariance estimation by linearly combining a ML estimate of each class covariance matrix with the ML pooled covariance estimate and with a scaled version of the identity matrix to form an estimate $\hat{\Sigma}_h(\lambda, \gamma)$ for the hth class:

$$\hat{\Sigma}_h(\lambda) = \frac{(1-\lambda)S_h + \lambda S}{(1-\lambda)n_h + \lambda n}, \tag{1}$$

$$\hat{\Sigma}_h(\lambda, \gamma) = (1-\gamma)\hat{\Sigma}_h(\lambda) + \frac{\gamma}{d}\,\mathrm{tr}\left(\hat{\Sigma}_h(\lambda)\right) I. \tag{2}$$

The above notation and other key notation is defined in Table 1.

In Friedman's regularized QDA (RDA), the parameters λ, γ are trained by crossvalidation to be those parameters that minimize the number of classification errors. Friedman's comparisons to ML QDA and ML linear discriminant analysis on six simulations showed that RDA could deal effectively with ill-posed covariance estimation when the true covariance matrix is diagonal. RDA is perhaps the most popular approach to discriminant analysis when the covariance estimation is expected to be ill-posed (Hastie et al., 2001).

Hoffbeck and Landgrebe (1996) proposed a similar regularized covariance estimate for classification of the form

$$\hat{\Sigma} = \alpha_1 \mathrm{diag}(\hat{\Sigma}_{ML}) + \alpha_2 \hat{\Sigma}_{ML} + \alpha_3 \hat{\Sigma}_{\mathrm{pooled}\,ML} + \alpha_4 \mathrm{diag}(\hat{\Sigma}_{\mathrm{pooled}\,ML}),$$

where $\hat{\Sigma}_{\mathrm{ML}}$ and $\hat{\Sigma}_{\mathrm{pooled}\,ML}$ are maximum likelihood estimates of class and pooled covariance matrices, respectively, and the parameters $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ are trained by crossvalidation to maximize the likelihood (whereas Friedman's RDA crossvalidates to maximize classification accuracy). Results on Friedman's simulation suite and experimental results on a hyperspectral classification problem showed that the two classifiers achieved similar accuracy (Hoffbeck and Landgrebe, 1996). Another restricted model used to regularize covariance matrix estimation is a banded covariance matrix (Bickel and Li, 2006).

RDA and the Hoffbeck-Landgrebe classifiers linearly combine different covariance estimates. A related approach is to select the best covariance model out of a set of models using crossvalidation. Bensmail and Celeux (1996) propose eigenvalue decomposition discriminant analysis (EDDA), in which fourteen different models for the class covariances are considered, ranging from a scalar times the identity matrix to a full class covariance estimate for each class. The model that minimizes the crossvalidation error is selected for use with the test data. Each individual model's parameters are estimated by ML; some of these estimates require iterative procedures that are computationally intensive.

The above techniques consider a discrete set of possible models for Σ, and either linearly combine models or select one model. In contrast, Bayesian QDA combines a continuous set of possible models for Σ, weighting each model by its posterior probability.

2.3 Other Approaches to Quadratic Discriminant Analysis

Other approaches have been developed for ill-posed quadratic discriminant analysis. Friedman (1989) notes that, beginning with work by James and Stein in 1961, researchers have attempted to improve the eigenvalues of the sample covariance matrix. Another approach is to reduce the data dimensionality before estimating the Gaussian distributions, for example by principal components analysis (Swets and Weng, 1996). One of the most recent algorithms of this type is orthogonal linear discriminant analysis (Ye, 2005), which was shown by the author of that work to perform similarly to Friedman's regularized linear discriminant analysis on six real data sets.

3. Distribution-based Bayesian Discriminant Analysis

Parameter estimation depends on the form of the parameter, for example, Bayesian estimation can yield one result if the expected standard deviation is solved for, or another result if the expected variance is solved for. To avoid this issue we derive Bayesian QDA by formulating the problem in terms of the Gaussian distributions explicitly. This section extends work presented in a recent conference paper (Srivastava and Gupta, 2006).

Suppose one is given an iid training set $\mathcal{T} = \{(x_i, y_i), i = 1, 2, \ldots n\}$ and a test sample x, where $x_i, x \in \mathbb{R}^d$, and y_i takes values from a finite set of class labels $y_i \in \{1, 2, \ldots, G\}$. Let C be the misclassification cost matrix such that $C(g,h)$ is the cost of classifying x as class g when the truth is class h. Let $P(Y = h)$ be the prior probability of class h. Suppose the true class conditional distributions $p(x|Y = h)$ exist and are known for all h, then the estimated class label for x that minimizes the expected misclassification cost is

$$Y^* \triangleq \underset{g=1,\ldots,G}{\operatorname{argmin}} \sum_{h=1}^{G} C(g,h) p(x|Y = h) P(Y = h). \tag{3}$$

In practice the class conditional distributions and the class priors are usually unknown. We model each unknown distribution $p(x|h)$ by a random Gaussian distribution N_h, and we model the unknown class priors by the random vector Θ, which has components $\Theta_h = P(Y = h)$ for $h = 1, \ldots, G$. Then, we estimate the class label that minimizes the expected misclassification cost, where the expectation is with respect to the random distributions Θ and $\{N_h\}$ for $h = 1, \ldots, G$. That is, define the *distribution-based Bayesian QDA* class estimate by replacing the unknown distributions in (3) with their random counterparts and taking the expectation:

$$\hat{Y} \triangleq \underset{g=1,\ldots,G}{\operatorname{argmin}} E\left[\sum_{h=1}^{G} C(g,h) N_h(x) \Theta_h\right]. \tag{4}$$

In (4) the expectation is with respect to the joint distribution over Θ and $\{N_h\}$ for $h = 1, \ldots, G$, and these distributions are assumed independent. Therefore (4) can be rewritten as

$$\hat{Y} = \underset{g=1,\ldots,G}{\operatorname{argmin}} \sum_{h=1}^{G} C(g,h) E_{N_h}[N_h(x)] E_\Theta[\Theta_h]. \tag{5}$$

Straightforward integration yields an estimate of the class prior, $E_\Theta[\Theta_h] = \frac{n_h+1}{n+G}$; this Bayesian estimate for the multinomial is also known as Laplace correction (Jaynes and Bretthorst, 2003).

In this next section we discuss the evaluation of $E_{N_h}[N_h(x)]$.

3.1 Statistical Models and Measure

Consider the family M of multivariate Gaussian probability distributions on $\mathbb{R}^d$. Let each element of M be a probability distribution $\mathcal{N} : \mathbb{R}^d \to [0,1]$, parameterized by the real-valued variables (μ, Σ) in some open set in $\mathbb{R}^d \otimes \mathbb{S}$, where $\mathbb{S} \subset \mathbb{R}^{d(d+1)/2}$ is the cone of positive semi-definite symmetric matrices. That is $M = \{\mathcal{N}(\cdot\, ; \mu, \Sigma)\}$ defines a $\frac{d^2+3d}{2}$-dimensional statistical model, (Amari and Nagaoka, 2000, pp. 25–28).

Let the differential element over the set M be defined by the Riemannian metric (Kass, 1989; Amari and Nagaoka, 2000),

$$\begin{aligned} dM &= |I_F(\mu,\Sigma)|^{\frac{1}{2}} d\mu d\Sigma, \quad \text{where} \\ I_F(\mu,\Sigma) &= -E_X[\nabla^2 \log \mathcal{N}(X; (\mu,\Sigma))], \end{aligned}$$

where ∇^2 is the Hessian operator with respect to the parameters μ and Σ, and this I_F is also known as the Fisher information matrix. Straightforward calculation shows that

$$dM = \frac{d\mu}{|\Sigma|^{\frac{1}{2}}} \frac{d\Sigma}{|\Sigma|^{\frac{d+1}{2}}} = \frac{d\mu d\Sigma}{|\Sigma|^{\frac{d+2}{2}}}. \tag{6}$$

Let $\mathcal{N}_h(\mu_h, \Sigma_h)$ be a possible realization of the Gaussian pdf N_h. Using the measure defined in (6),

$$E_{N_h}[N_h(x)] = \int_M \mathcal{N}_h(x) r(\mathcal{N}_h) dM, \tag{7}$$

where $r(\mathcal{N}_h)$ is the posterior probability of $\mathcal{N}_h$ given the set of class h training samples $\mathcal{T}_h$; that is,

$$r(\mathcal{N}_h) = \frac{\ell(\mathcal{N}_h, \mathcal{T}_h) p(\mathcal{N}_h)}{\alpha_h}, \tag{8}$$

where α_h is a normalization constant, $p(\mathcal{N}_h)$ is the prior probability of $\mathcal{N}_h$ (treated further in Section 3.2), and $\ell(\mathcal{N}_h, \mathcal{T}_h)$ is the likelihood of the data $\mathcal{T}_h$ given $\mathcal{N}_h$, that is,

$$\ell(\mathcal{N}_h(\mu_h, \Sigma_h), \mathcal{T}_h) = \frac{\exp[-\frac{1}{2}\operatorname{tr}\left(\Sigma_h^{-1} S_h\right) - \frac{n_h}{2}\operatorname{tr}\left(\Sigma_h^{-1}(\mu_h - \bar{X}_h)(\mu_h - \bar{X}_h)^T\right)]}{(2\pi)^{\frac{dn_h}{2}} |\Sigma_h|^{\frac{n_h}{2}}}. \tag{9}$$

3.2 Priors

A prior probability distribution of the Gaussians, $p(\mathcal{N}_h)$, is needed to solve the classification problem given in (4). A common interpretation of Bayesian analysis is that the prior represents information that one has prior to seeing the data (Jaynes and Bretthorst, 2003). In the practice of statistical learning, one often has very little quantifiable information apart from the data. Instead of thinking of the prior as representing prior information, we considered the following design goals: the prior should

- regularize the classification to reduce estimation variance, particularly when the number of training samples n is small compared to the number of feature dimensions;
- add minimal bias;

- allow the estimation to converge to the true generating class conditional normals as $n \to \infty$ if in fact the data was generated by class conditional normals;
- lead to a closed-form result.

To meet these goals, we use as a prior

$$p(\mathcal{N}_h) = p(\mu_h)p(\Sigma_h) = \gamma_0 \frac{\exp[-\frac{1}{2}\operatorname{tr}\left(\Sigma_h^{-1}B_h\right)]}{|\Sigma_h|^{\frac{q}{2}}}, \tag{10}$$

where B_h is a positive definite matrix (further specified in (25)) and γ_0 is a normalization constant. The prior (10) is equivalent to a noninformative prior for the mean μ, and an inverted Wishart prior with q degrees of freedom over Σ.

This prior is unimodal and leads to a closed-form result. Depending on the choice of B_h and q, the prior probability mass can be focused over a small region of the set of Gaussian distributions M in order to regularize the estimation. Regularization is important for cases where the number of training samples n is small compared to the dimensionality d. However, the tails of this prior are sufficiently heavy that the prior does not hinder convergence to the true generating distribution as the number of training samples increases if the true generating distribution is normal.

The positive definite matrix B_h specifies the location of the maximum of the prior probability distribution. Using the matrix derivative (Dwyer, 1967) and the knowledge that the inverse Wishart distribution has only one maximum, one can calculate the location of the maximum of the prior. Take the log of the prior,

$$\log p(\mathcal{N}_h) = -\frac{1}{2}\operatorname{tr}\left(\Sigma_h^{-1}B_h\right) - \frac{q}{2}\log|\Sigma_h| + \log\gamma_0.$$

Differentiate with respect to Σ_h to solve for $\Sigma_{h,max}$,

$$\begin{aligned} -\frac{1}{2}\frac{\partial}{\partial\Sigma_h}\operatorname{tr}\left(\Sigma_{h,max}^{-1}B_h\right) - \frac{q}{2}\frac{\partial}{\partial\Sigma_h}\log|\Sigma_{h,max}| &= 0, \\ \Sigma_{h,max}^{-1}B_h\Sigma_{h,max}^{-1} - q\Sigma_{h,max}^{-1} &= 0, \\ \Sigma_{h,max} &= \frac{B_h}{q}. \end{aligned} \tag{11}$$

Because this prior is unimodal, a rough interpretation of its action is that it regularizes the likelihood covariance estimate towards the maximum of the prior, given in (11). To meet the goal of minimizing bias, we encode some coarse information about the data into B_h. In QB (Brown et al., 1999), the prior seed matrix $B_h = kI$, where k is a scalar determined by crossvalidation. Setting $B_h = kI$ is reminiscent of Friedman's RDA (Friedman, 1989), where the covariance estimate is regularized by the trace: $\frac{\operatorname{tr}\left(\hat{\Sigma}_{ML}\right)}{d}I$.

We have shown in earlier work that setting $B_h = \frac{\operatorname{tr}\left(\hat{\Sigma}_{ML}\right)}{d}I$ for a distribution-based discriminant analysis outperforms Geisser's or Keehn's parameter-based Bayesian discriminant methods, and does not require crossvalidation (Srivastava and Gupta, 2006). The trace of the ML covariance estimate is stable, and provides coarse information about the scale of the data samples. Thus, this proposed data-dependent prior can be interpreted as capturing the knowledge an application expert would have before seeing the actual data. There are many other approaches to data-dependent priors, including hyperparameters and empirical Bayes methods (Efron and Morris, 1976; Haff, 1980; Lehmann and Casella, 1998). Data-dependent priors are also used to form proper priors that act similarly to improper priors, or to match frequentist goals (Wasserman, 2000). Next, we describe the closed form result with a prior of the form given in (10), then we return to the question of data-dependent definitions for B_h when we propose the BDA7 classifier in Section 6.

3.3 Closed-Form Result

In Theorem 1 we establish the closed-form result for the distribution-based Bayesian QDA classifier. The closed-form result for the parameter-based classifier with the same prior is presented after the theorem for comparison.

Theorem 1: The classifier (5) using the prior (10) can be written as

$$\hat{Y} = \underset{g=1,\ldots,G}{\operatorname{argmin}} \sum_{h=1}^{G} C(g,h) \frac{(n_h)^{\frac{d}{2}} \Gamma\left(\frac{n_h+q+1}{2}\right) \left|\frac{S_h+B_h}{2}\right|^{\frac{n_h+q}{2}}}{(n_h+1)^{\frac{d}{2}} \Gamma\left(\frac{n_h+q-d+1}{2}\right) |A_h|^{\frac{n_h+q+1}{2}}} \hat{P}(Y=h), \tag{12}$$

where $\hat{P}(Y=h)$ is an estimate of the class prior probability for class h, $\Gamma(\cdot)$ is the standard gamma function, and

$$A_h = \frac{1}{2}\left(S_h + \frac{n_h(x-\bar{x}_h)(x-\bar{x}_h)^T}{(n_h+1)} + B_h\right). \tag{13}$$

The proof of the theorem is given in Appendix A. The parameter-based Bayesian QDA class label using the prior given in (10) is

$$\hat{Y} = \underset{g=1,\ldots,G}{\operatorname{argmin}} \sum_{h=1}^{G} C(g,h) \frac{(n_h)^{\frac{d}{2}} \Gamma\left(\frac{n_h+q-d-1}{2}\right)}{(n_h+1)^{\frac{d}{2}} \Gamma\left(\frac{n_h+q-2d-1}{2}\right)} \frac{\left|\frac{S_h+B_h}{2}\right|^{\frac{n_h+q-d-2}{2}}}{|A_h|^{\frac{n_h+q-d-1}{2}}} \hat{P}(Y=h). \tag{14}$$

Equation (14) can be proved by following the same steps as the proof of Theorem 1. The parameter-based Bayesian discriminant result (14) will not hold if $n_h \leq 2d - q + 1$, while the distribution-based result (12) holds for any $n_h > 0$ and any d.

In a previous publication (Srivastava and Gupta, 2006), we compared the distribution-based Bayesian QDA to the parameter-based Bayesian QDA classifier, maximum likelihood QDA, and nearest means. We compared the noninformative prior originally proposed by Geisser (1964), and the modified inverse Wishart prior given in Equation (10) with $B_h = \operatorname{tr}\left(\hat{\Sigma}_{ML,h}/d\right) I$. We fixed the degrees of freedom for the modified inverse Wishart to be $d+3$, so that in one-dimension it reduces to the common inverted gamma distribution. Results on six simulations (two-class versions of Friedman's original simulations) showed that given the same prior, the distribution-based performed better than the parameter-based when the ratio of feature dimensions to number of training samples is high. For the reader's convenience, we include a representative set of the simulation results from Srivastava and Gupta (2006) in Figure 1.

It is interesting to see that the only difference between the parameter-based Bayesian QDA (14) and the distribution-based Bayesian QDA (12) is a shift of the degree of freedom q; this is true whether the distribution-based formula (12) is solved for using either the Fisher or Lebesgue measure. QB, which is a modern parameter-based Bayesian QDA classifier (discussed further in Section

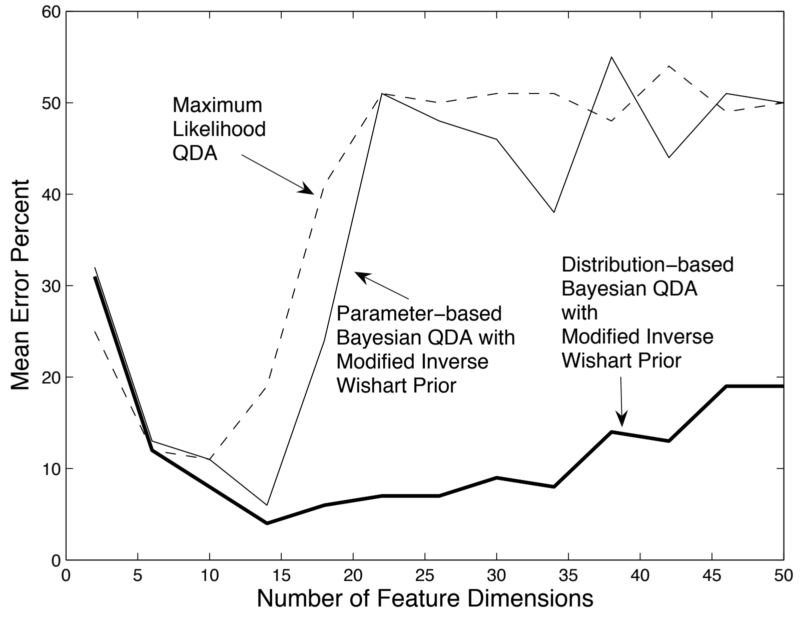


Figure 1: Mean error rates are shown for a two-class simulation where the samples from each class are drawn from a Gaussian distribution with the same mean but different, highly-ellipsoidal covariance matrices. The error rates were averaged over 1000 random trials, where on each trial 40 training samples and 100 test samples were drawn iid.

2.1), chooses the degree of freedom for the modified inverse Wishart prior by cross-validation. If one cross-validates the degree of freedom, then it does not matter if one starts from the parameter-based formula or the distribution-based formula. In Section 6, we propose a new Bayesian QDA classifier called *BDA7* that cross-validates the degree of freedom and a data-dependent seed matrix B_h for the prior. But first we consider further the analytic properties of Bayesian QDA; in the next section we develop a relationship between regularized QDA and Bayesian QDA, and then in Section 5 we show how Bayesian QDA is the solution to minimizing any expected Bregman divergence.

4. Relationship Between Regularized QDA and Bayesian QDA

In this section we show that Friedman's regularized form for the covariance matrix (Friedman, 1989) emerges from the Bayesian QDA formula.

Let $D_h = S_h + B_h, Z_h = x - \bar{x}_h$. The distribution-based Bayesian discriminant formula for the class conditional pdf (12) can be simplified to

$$\begin{aligned}
E_{N_h}[N_h] &= \frac{n_h^{\frac{d}{2}}\Gamma\left(\frac{n_h+q+1}{2}\right)}{(2\pi)^{\frac{d}{2}}(n_h+1)^{\frac{d}{2}}\Gamma\left(\frac{n_h+q-d+1}{2}\right)} \frac{\left|\frac{D_h}{2}\right|^{\frac{n_h+q}{2}}}{\left|\frac{D_h}{2}+\frac{n_h}{2(n_h+1)}Z_h Z_h^T\right|^{\frac{n_h+q+1}{2}}} \\
&= \frac{n_h^{\frac{d}{2}}\Gamma\left(\frac{n_h+q+1}{2}\right)\left|\frac{D_h}{2}\right|^{\frac{n_h+q}{2}}}{(2\pi)^{\frac{d}{2}}(n_h+1)^{\frac{d}{2}}\Gamma\left(\frac{n_h+q-d+1}{2}\right)\left|\frac{D_h}{2}\right|^{\frac{n_h+q+1}{2}}} \left|I+\frac{n_h}{n_h+1}Z_h Z_h^T D_h^{-1}\right|^{-\frac{n_h+q+1}{2}} \\
&= \frac{\Gamma\left(\frac{n_h+q+1}{2}\right)}{(\pi)^{\frac{d}{2}}\left|\left(\frac{n_h+1}{n_h}\right)D_h\right|^{\frac{1}{2}}\Gamma\left(\frac{n_h+q-d+1}{2}\right)}\left(1+\frac{n_h}{n_h+1}Z_h^T D_h^{-1} Z_h\right)^{-\frac{n_h+q+1}{2}}, \qquad (15)
\end{aligned}$$

where (15) follows by rearranging terms and applying the identity $|I + Z_h Z_h^T D_h^{-1}| = 1 + Z_h^T D_h^{-1} Z_h$ (Anderson, 2003).

Approximate $n_h/(n_h+1) \approx 1$ in (15). Recall the series expansion $e^r = 1 + r + r^2/2\ldots$, so if r is small, $1 + r \approx e^r$. We apply this approximation to the term $1 + Z_h^T D_h^{-1} Z_h$ in (15), and note that the approximation is better the closer the test point x is to the sample mean $\bar{x}_h$, such that Z_h is small. The approximation is also better the larger the minimum eigenvalue λ_{min} of D_h is, because of the bound $|Z_h^T D_h^{-1} Z_h| \leq \|Z_h\|^2/\lambda_{min}$. Then (15) becomes

$$\begin{aligned}
E_{N_h}[N_h] &\approx \frac{\Gamma\left(\frac{n_h+q+1}{2}\right)\left(\exp\left[\frac{n_h}{n_h+1}Z_h^T D_h^{-1} Z_h\right]\right)^{-\frac{n_h+q+1}{2}}}{(\pi)^{\frac{d}{2}}\left|\left(\frac{n_h+1}{n_h}\right)D_h\right|^{\frac{1}{2}}\Gamma\left(\frac{n_h+q-d+1}{2}\right)}, \\
&= \frac{\Gamma\left(\frac{n_h+q+1}{2}\right)\exp\left[-\frac{1}{2}Z_h^T\left[\frac{n_h+1}{n_h+q+1}\left(\frac{D_h}{n_h}\right)\right]^{-1} Z_h\right]}{(\pi)^{\frac{d}{2}}\left|\left(\frac{n_h+1}{n_h}\right)D_h\right|^{\frac{1}{2}}\Gamma\left(\frac{n_h+q-d+1}{2}\right)}. \qquad (16)
\end{aligned}$$

Let

$$\tilde{\Sigma}_h \triangleq \frac{n_h+1}{n_h+q+1}\frac{D_h}{n_h}. \qquad (17)$$

The approximation (16) resembles a Gaussian distribution, where $\tilde{\Sigma}_h$ plays the role of the covariance matrix. Rewrite (17),

$$\begin{aligned}
\tilde{\Sigma}_h &= \frac{n_h+1}{n_h+q+1}\left(\frac{S_h+B_h}{n_h}\right) \\
&= \frac{n_h+1}{n_h+q+1}\left(\frac{S_h}{n_h}\right) + \frac{n_h+1}{n_h+q+1}\left(\frac{B_h}{n_h}\right) \\
&= \left(1-\frac{q}{n_h+q+1}\right)\frac{S_h}{n_h} + \frac{1}{n_h+q+1}\left(\frac{n_h+1}{n_h}\right)B_h. \qquad (18)
\end{aligned}$$

In (18), make the approximation $\frac{n_h+1}{n_h} \approx 1$, then multiply and divide the second term of (18) by q,

$$\tilde{\Sigma}_h \approx \left(1-\frac{q}{n_h+q+1}\right)\frac{S_h}{n_h} + \left(\frac{q}{n_h+q+1}\right)\frac{B_h}{q}. \qquad (19)$$

The right-hand side of (19) is a convex combination of the sample covariance and the positive definite matrix $\frac{B_h}{q}$. This is the same general formulation as Friedman's RDA regularization (Friedman, 1989), re-stated in this paper in Equations (1) and (2). Here, the fraction $\frac{q}{n_h+q+1}$ controls the shrinkage of the sample covariance matrix toward the positive definite matrix $\frac{B_h}{q}$; recall from (11) that $\frac{B_h}{q}$ is the maximum of the prior probability distribution. Equation (19) also gives information about how the Bayesian shrinkage depends on the number of sample points from each class: fewer training samples n_h results in greater shrinkage towards the positive definite matrix $\frac{B_h}{q}$. Also, as the degree of freedom q increases, the shrinkage towards $\frac{B_h}{q}$ increases. However, as q increases, the shrinkage target $\frac{B_h}{q}$ moves towards the zero-matrix.

5. Bregman Divergences and Bayesian Quadratic Discriminant Analysis

In (5) we defined the Bayesian QDA class estimate that minimizes the expected misclassification cost. Then assuming a Gaussian class-conditional distribution, the expected class-conditional distribution is given in (12). A different approach to Bayesian estimation would be to estimate the hth class-conditional distribution to minimize some expected risk. That is, the estimated class-conditional distribution would be

$$\hat{f}_h = \operatorname*{argmin}_{f \in \mathcal{A}} \int_M R(\mathcal{N}_h, f)\, dM, \tag{20}$$

where $R(\mathcal{N}_h, f)$ is the risk of guessing f if the truth is $\mathcal{N}_h$, the set of functions $\mathcal{A}$ is defined more precisely shortly, and dM is a probability measure on the set of Gaussians, M. Equation (20) is a distribution-based version of the standard parameter Bayesian estimate given in Ch. 4 of Lehmann and Casella (1998); for example, the standard parameter Bayesian estimate of the mean $\hat{\mu} \in \mathbb{R}^d$ would be formulated

$$\hat{\mu} = \operatorname*{argmin}_{\psi \in \mathbb{R}^d} \int R(\mu, \psi)\, d\Lambda(\mu),$$

where $\Lambda(\mu)$ is some probability measure.

Given estimates of the class-conditional distributions $\{\hat{f}_h\}$ from (20), one can solve for the class label as

$$\tilde{Y}^* = \operatorname*{argmin}_{g=1,\dots,G} \sum_{h=1}^{G} C(g,h) \hat{f}_h(x) \hat{P}(Y=h). \tag{21}$$

In this section we show that the class estimate $\hat{Y}$ from minimizing the expected misclassification cost as defined in (5) is equivalent to the class estimate $\tilde{Y}^*$ from (21) if the risk function in (20) is a (functional) Bregman divergence. This result links minimizing expected misclassification cost and minimizing an expected Bregman divergence.

Bregman divergences form a set of distortion functions that include squared error, relative entropy, logistic loss, Mahalanobis distance, and the Itakura-Saito function, and are sometimes termed *Bregman loss functions* (Censor and Zenios, 1997). Bregman divergences act on pairs of vectors. Csiszár defined a Bregman divergence between two distributions (Csiszár, 1995), but Csiszár's definition acts pointwise on the input distributions, which limits its usefulness in analysis. A recent result showed that the mean minimizes the average Bregman divergence (Banerjee et al., 2005a,b). In order to extend this result to distributions and show how it links to Bayesian estimation, one must solve for minima over sets of functions. To this end, we define a new *functional Bregman* *divergence* that acts on pairs of distributions. This allows us to extend the Banerjee et al. result to the Gaussian case and establish the equivalence between minimizing expected misclassification cost and minimizing the expected functional Bregman divergence.

This section makes use of functional analysis and the calculus of variations; the relevant definitions and results from these fields are provided for reference in Appendix B.

Let ν be some measure, and define the set of functions $\mathcal{A}_p$ to be

$$\mathcal{A}_p = \left\{ a : \mathbb{R}^d \to [0,1] \;\middle|\; a \in L^p(\nu),\, a > 0,\, \|a\|_{L^p(\nu)} = 1 \right\}.$$

5.1 Functional Definition of Bregman Divergence

Let $\phi : \mathcal{A}_p \to \mathbb{R}$ be a continuous functional. Let $\delta^2\phi[f;\cdot,\cdot]$, the second variation of ϕ, be strongly positive. The functional Bregman divergence $d_\phi : \mathcal{A}_p \times \mathcal{A}_p \to [0,\infty)$ is defined as

$$d_\phi(f,g) = \phi[f] - \phi[g] - \delta\phi[g; f-g], \tag{22}$$

where $\delta\phi[g;\cdot]$ is the Fréchet derivative or first variation of ϕ at g.

Different choices of the functional ϕ will lead to different Bregman divergences. As an example, we present the ϕ for squared error.

5.2 Example (Squared error)

Let $\phi : \mathcal{A}_2 \to \mathbb{R}$ be defined

$$\phi[g] = \int g^2 d\nu.$$

Perturbing g by a sufficiently nice function a (see Appendix B for more details about the perturbation function a) leads to the difference

$$\phi[g+a] - \phi[g] = \int \left(g^2 + 2ga + a^2 - g^2\right) d\nu.$$

Then, because

$$\begin{aligned} \frac{\|\phi[g+a] - \phi[g] - \int 2ga d\nu\|_{L^2(\nu)}}{\|a\|_{L^2(\nu)}} &= \frac{\int a^2\, d\nu}{(\int a^2\, d\nu)^{\frac{1}{2}}}, \\ &= \left(\int a^2\, d\nu\right)^{\frac{1}{2}} \end{aligned}$$

tends to zero as $\|a\|_{L^2(\nu)}$ tends to zero, the differential is

$$\delta\phi[g;a] = \int 2ga\, d\nu. \tag{23}$$

Using (23), the Bregman divergence (22) becomes

$$\begin{aligned} d_\phi(f,g) &= \int f^2 d\nu - \int g^2 d\nu - \int 2g(f-g)d\nu \\ &= \int f^2 d\nu - \int 2fg d\nu + \int g^2 d\nu \\ &= \int (f-g)^2 d\nu \\ &= \|f-g\|^2_{L^2(\nu)}, \end{aligned}$$

which is the integrated squared error between two functions f and g in $\mathcal{A}_2$.

5.3 Minimizing Expected Bregman Divergence

The functional definition (22) is a generalization of the standard vector Bregman divergence

$$d_{\tilde\phi}(x,y) = \tilde\phi(x) - \tilde\phi(y) - \nabla\tilde\phi(y)^T(x-y)$$

where $x, y \in \mathbb{R}^n$, and $\tilde\phi : \mathbb{R}^n \to \mathbb{R}$ is strictly convex and twice differentiable.

Theorem 2: Let $\phi : \mathcal{A}_1 \to \mathbb{R}$, $\phi \in \mathcal{C}^3$, and $\delta^2\phi[f;\cdot,\cdot]$ be strongly positive. Suppose the function $\hat{f}_h$ minimizes the expected Bregman divergence between a random Gaussian N_h and any probability density function $f \in \mathcal{A}$ where the expectation is taken with respect to the distribution $r(\mathcal{N}_h)$, such that

$$\hat{f}_h = \underset{f\in\mathcal{A}}{\operatorname{argmin}} E_{N_h}[d_\phi(N_h, f)]. \tag{24}$$

Then $\hat{f}_h$ is given by

$$\hat{f}_h = \int_M \mathcal{N}_h\, r(\mathcal{N}_h) dM = E_{N_h}[N_h(x)].$$

The proof of the theorem is given in Appendix A.

Corollary: The result of (5) is equivalent to the result of (21) where each $\hat{f}_h$ comes from (24).

The corollary follows directly from Theorem 2 where $r(\mathcal{N}_h)$ is the posterior distribution of $\mathcal{N}_h$ given the training samples.

6. The BDA7 Classifier

Distribution-based QDA with a fixed degree of freedom as proposed by the authors in the conference paper (Srivastava and Gupta, 2006) does not require cross-validation. However, with cross-validation, one can generally do better if cross-validating a useful parameter. The question is, what parameters to cross-validate with what options? As done in the QB Bayesian QDA classifier, we believe that the degree of freedom of the prior should be cross-validated. Also, in preliminary experiments we found that the distribution-based performance could be enhanced by using the diagonal of $\hat\Sigma_{ML}$ rather than the trace for each prior seed matrix B_h; the diagonal encodes more information but is still relatively stable to estimate. The diagonal of $\hat{\Sigma}_{ML}$ has been used to regularize a regularized discriminant analysis (Hoffbeck and Landgrebe, 1996) and model-based discriminant analysis (Bensmail and Celeux, 1996).

Note that setting $B_h = \text{diag}\left(\hat{\Sigma}_{\text{ML}}\right)$ places the maximum of the prior at $\frac{1}{q}\text{diag}\left(\hat{\Sigma}_{\text{ML}}\right)$. We hypothesize that in some cases it may be more effective to place the maximum of the prior at diag $\left(\hat{\Sigma}_{\text{ML}}\right)$; that requires setting $B_h = q\ \text{diag}\left(\hat{\Sigma}_{\text{ML}}\right)$.

We also consider moving the maximum of the prior closer to the zero matrix, effectively turning the prior into an exponential prior rather than a unimodal one. We expect that this will have the rough effect of shrinking the estimate toward zero. Shrinkage towards zero is a successful technique in other estimation scenarios: for example ridge and lasso regression shrink linear regression coefficients toward zero (Hastie et al., 2001), and wavelet denoising shrinks wavelet coefficients toward zero. To this end, we also consider setting the prior matrix seed to be $B_h = \frac{1}{q}\text{diag}\left(\hat{\Sigma}_{\text{ML}}\right)$.

These different choices for B_h will be more or less appropriate depending on the amount of data and the true generating distributions. Thus, for BDA7 the B_h is selected by crossvalidation from seven options:

$$B_h = \begin{cases} q\ \text{diag}\left(\hat{\Sigma}_{(\text{pooled ML})}\right),\ \text{q diag}\left(\hat{\Sigma}_{(\text{class ML,h})}\right), \\ \frac{1}{q}\ \text{diag}\left(\hat{\Sigma}_{(\text{pooled ML})}\right),\ \frac{1}{\text{q}}\ \text{diag}\left(\hat{\Sigma}_{(\text{class ML,h})}\right), \\ \text{diag}\left(\hat{\Sigma}_{(\text{pooled ML})}\right),\ \text{diag}\left(\hat{\Sigma}_{(\text{class ML,h})}\right), \\ \frac{1}{q}\,\text{tr}(\hat{\Sigma}_{(\text{pooled ML})})\ I. \end{cases} \tag{25}$$

To summarize: BDA7 uses the result (12) where q is cross-validated, and B_h is cross-validated as per (25).

7. Experiments with Benchmark Data Sets

We compared BDA7 to popular QDA classifiers on ten benchmark data sets from the UCI Machine Learning Repository. In the next section we use simulations to further analyze the behavior of each of the classifiers.

QDA classifiers are best-suited for data sets where there is relatively little data, assuming the class-conditional distribution is Gaussian model can be an appropriate model. For this reason, for data sets with separate training and test sets, Tables 3 and 4 show results on the test set given a randomly chosen 5% or 10% of the training samples. For data sets without separate training and test sets, 5% and 10% of the available samples of each class were randomly selected and used for training, and the rest of the available samples were used as test samples. Each result in Table 3 and 4 is the average test error of 100 trials with different randomly chosen training samples, with the exception of the Cover Type data set, for which only 10 random trials were performed due to its large size. Table 2 summarizes each of the benchmark data sets.

7.1 Experimental Details for Each Classifier

BDA7 is compared with QB (Brown et al., 1999), RDA (Friedman, 1989), eigenvalue decomposition discriminant analysis (EDDA) (Bensmail and Celeux, 1996), and maximum-likelihood estimated QDA, LDA, and the nearest-means classifier (NM). Code for the classifiers (and the simulations presented in the next section) is available at idl.ee.washington.edu.

The parameters for each classifier were estimated by leave-one-out crossvalidation, unless there exists a separate validation set. Some of the classifiers required the prior probability of each class

$P(Y = h)$; those probabilities were estimated based on the number of observations from each class using Bayesian estimation (see Section 3).

The RDA parameters λ and γ were calculated and crossvalidated as in Friedman's paper (Friedman, 1989) for a total of 25 joint parameter choices.

The BDA7 method is crossvalidated with the seven possible choices of B_h for the prior specified in (25). The scale parameter q of the inverse Wishart distribution is crossvalidated in steps of the feature space dimension d so that $q \in \{d, 2d, 3d, 4d, 5d, 6d\}$. Thus there are 42 parameter choices.

The QB method is implemented as described by Brown et al. (1999). QB uses a normal prior for each mean vector, and an inverse Wishart distribution prior for each covariance matrix. There are two free parameters to the inverse Wishart distribution: the scale parameter $q \geq d$ and the seed matrix B_h. For QB, B_h is restricted to be spherical: $B_h = kI$. The parameters q and k are trained by crossvalidation. In an attempt to be similar to the RDA and BDA7 crossvalidations, we allowed there to be 42 parameter choices, $q \in \{d, 2d, 3d, 4d, 5d, 6d\}$ and $k \in \{1, 2, \ldots 7\}$. Other standard Bayesian quadratic discriminant approaches use a uniform (improper) or fixed Wishart prior with a parameter-based classifier (Ripley, 2001; Geisser, 1964; Keehn, 1965); we have previously demonstrated that the inverse Wishart prior performs better than these other choices (Srivastava and Gupta, 2006).

The EDDA method of Bensmail and Celeux is run as proposed in their paper (Bensmail and Celeux, 1996). Thus, the crossvalidation selects one of fourteen models for the covariance, where the ML estimate for each model is used. Unfortunately, we found it computationally infeasible to run the 3rd and 4th model for EDDA for some of the problems. These models are computationally intensive iterative ML procedures that sometimes took prohibitively long for large n, and when $n < d$ these models sometimes caused numerical problems that caused our processor to exit with error. Thus, in cases where the 3rd and 4th model were not feasible, they were not used.

7.2 Results

The results are shown in Tables 3 and 4 for the UCI Machine Learning Repository benchmark data sets described in Table 2. The lowest mean error rate is in bold, as well as any other mean error rate that is not statistically significantly different from the lowest mean error rate, as determined by the Wilcoxon signed rank test for paired-differences at a significance level of .05.

BDA7 was the best or statistically insignificant from the best for seven of the ten tested data sets: Heart, Iris, Image Segmentation, Pen Digits, Sonar, Thyroid, and Wine. For the Pima Diabetes, Ionosphere, and Cover Type data sets, BDA7 is not the best QDA classifier, but performs competitively.

The Ionosphere data set is a two-class problem with 351 samples described by 34 features. One of the features has the value zero for all samples of class two. This causes numerical difficulties in estimating the maximum likelihood covariance estimates. BDA7 chooses the identity prior seed matrix $B_h = \frac{1}{q}\operatorname{tr}(\hat{\Sigma}_{(\text{pooled ML})})\, I$ every time for this data set. Given that this is BDA7's choice, BDA7 is at a disadvantage compared to QB, which cross-validates a scaled identity seed matrix kI. In fact, our experiments have showed that shrinking the prior closer to zero improves the performance on this data set, for example using $B_h = \frac{1}{qd}\operatorname{tr}(\hat{\Sigma}_{(\text{pooled ML})})\, I$.

For the Cover Type problem, the maximum likelihood estimated matrices for QDA and LDA contain many zeros, which result in a zero determinant; this is in part because 44 of the 54 features are binary. Thus the QDA and LDA error rate is solely due to the prior class probabilities. Because of the zero determinant, the best BDA7 model for the prior's seed matrix is the identity-based model,

	Number of Classes	Number of Features	Number of Total Samples (Training/Test) (Training/Validation/Test)
Cover Type	8	54	11,340/3,780/565,892
Heart	2	10 (nominal features excluded)	270
Image Segmentation	7	19	2310
Ionosphere	2	34	351
Iris	3	4	150
Pen Digits	10	16	7,494/3,498
Pima	2	8	768
Sonar	2	60	238
Thyroid	3	5	215
Wine	3	13	178

Table 2: Information About the Benchmark Data Sets

	BDA7	QB	RDA	EDDA	NM	LDA	QDA
Cover Type	48.3	**44.6**	59.2	83.1	78.5	62.9	62.9
Heart	**30.6**	38.5	38.2	33.9	39.9	35.9	44.5
Image Segmentation	**12.7**	**12.9**	18.3	29.7	29.5	85.7	85.7
Ionosphere	16.9	16.1	**12.5**	26.0	27.1	35.8	35.8
Iris	**6.9**	**7.6**	**8.1**	9.4	9.3	8.8	66.6
Pen Digits	**6.1**	8.1	8.5	7.7	23.9	17.8	23.8
Pima	29.7	32.7	29.4	30.7	35.8	**27.6**	33.4
Sonar	**36.8**	40.4	40.4	39.8	42.6	46.7	46.7
Thyroid	**11.7**	14.8	17.0	14.7	19.2	15.0	30.0
Wine	**9.6**	33.1	34.2	**11.2**	32.0	60.1	60.1

Table 3: Average Error Rate Using Randomly Selected 5% of Training Samples

	BDA7	QB	RDA	EDDA	NM	LDA	QDA
Cover Type	48.0	**43.7**	57.4	82.1	77.9	62.9	62.9
Heart	**27.4**	32.0	31.8	**28.3**	38.6	**28.1**	41.8
Image Segmentation	**10.9**	**10.8**	16.4	27.9	27.7	85.7	85.7
Ionosphere	12.5	11.1	**8.7**	23.3	24.2	35.9	35.9
Iris	**6.2**	**5.9**	**6.2**	7.4	8.3	**5.3**	44.9
Pen Digits	**5.3**	6.0	7.5	5.6	22.9	17.3	16.9
Pima	28.4	29.7	27.7	29.0	36.1	**25.8**	29.8
Sonar	**31.2**	33.7	32.8	34.8	38.8	46.8	46.8
Thyroid	**7.9**	9.1	10.0	**8.6**	16.2	10.5	30.0
Wine	**7.9**	16.9	25.0	**8.2**	30.1	21.6	60.3

Table 4: Average Error Rate Using Randomly Selected 10% of Training Samples

$B_h = \frac{1}{q}\,\mathrm{tr}(\hat{\Sigma}_{(\text{pooled ML})})\, I$. As in the Ionosphere data set, this puts BDA7 at a disadvantage compared to QB, and it is not surprising that on this data set QB does a little better. Despite the numerical difficulties inherent in this problem, the two Bayesian QDA classifiers do better than the other QDA classifiers.

8. Simulations

In order to further analyze the behavior of the different QDA classifiers, we compared them in ten simulations. For each of the ten simulations, the data are drawn iid from three Gaussian class conditional distributions. Six of the simulations were originally created by Friedman and used to show that RDA performs favorably compared to ML linear discriminant analysis (LDA) and ML QDA (Friedman, 1989). Friedman's six simulations all have diagonal generating class covariance matrices, corresponding to independent classification features. In those cases, the constrained diagonal models used in RDA and EDDA are correct, and so RDA and EDDA's performance may be optimistic compared to real data. For a fuller picture, we add two full covariance matrix simulations to Friedman's six diagonal Gaussian simulations (Cases 1–6), and for each of the full covariance matrix simulations we consider the cases of classes with the same means (Case 7 and 9), and classes with different means (Case 8 and 10).

All of the simulation results are presented for 40 training and 100 test samples, drawn iid. The parameters for each classifier were estimated by leave-one-out crossvalidation, as described in Section 7.1. Each simulation was run 100 times. Thus, each result presented in Tables 5, 6, and 7 is the average error over $10,000$ test samples.

8.1 Simulation Details and Results

The results show that when the true generating distributions are diagonal (Cases 1–6), then EDDA performs the best. This can be expected because many of the 14 models EDDA cross-validates are diagonal, and are thus a good match to the true generating distributions. When the true generating distributions are drawn from full covariance matrices (Cases 7–10), EDDA performs well when there are only 10 features, but the maximum likelihood estimates used in EDDA fail for 50 and 100 feature dimensions. RDA similarly does well when the true generating distribution matches the RDA models, and like EDDA is unable to learn when the features are highly correlated (Cases 9–10).

BDA7 is rarely the best classifier on the simulations, but never fails to produce relatively reasonable error rates. In particular, BDA7 is shown to be a more robust classifier than QB, achieving much lower error rates for Cases 5–6, and lower error rates for Cases 2–4 (with the exception of Case 4 for 100 feature dimensions).

A more detailed analysis and description of each of the simulation cases follows.

8.1.1 CASE 1: EQUAL SPHERICAL COVARIANCE MATRICES

Each class conditional distribution is normal with identity covariance matrix I. The mean of the first class μ_1 is the origin, and the second class has zero mean, except that the first component of the second class mean is 3. Similarly, the third class has zero mean, except the last component of the third class mean is 3.

The performance here is bounded by the nearest-means classifier, which is optimal for this simulation. EDDA also does well, because one of the 14 models available for it to choose is exactly correct: scalar times the identity matrix. Similarly, RDA strongly shrinks towards the trace times the identity. Though this is an unrealistic case, it shows that BDA7 can perform relatively well even though it does not explicitly model the true generating distribution.

8.1.2 CASE 2: UNEQUAL SPHERICAL COVARIANCE MATRICES

The class one conditional distribution is normal with identity covariance matrix I and mean at the origin. The class two conditional distribution is normal with covariance matrix $2I$ and has zero mean except the first component of its mean is 3. The class three conditional distribution is normal with covariance matrix $3I$ and has zero mean except the last component of its mean is 4. Like Case 1, EDDA and RDA use the fact that the true generating distributions match their models to outperform BDA7, which is significantly better than QB.

8.1.3 CASES 3 AND 4: EQUAL HIGHLY ELLIPSOIDAL COVARIANCE MATRICES

Covariance matrices of each class distribution are the same, and highly ellipsoidal. The eigenvalues of the common covariance matrix are given by

$$e_i = \left(\frac{9(i-1)}{d-1}+1\right)^2, \quad 1 \le i \le d, \tag{26}$$

so the ratio of the largest to smallest eigenvalue is 100.

For Case 3 the class means are concentrated in a low-variance subspace. The mean of class one is located at the origin and the i^{th} component of the mean of class two is given by

$$\mu_{2i} = 2.5\sqrt{\frac{e_i}{d}}\frac{d-i}{(\frac{d}{2}-1)}, \quad 1 \le i \le d.$$

	BDA7	QB	RDA	EDDA	NM	LDA	QDA
Case 1	13.2	19.2	12.4	11.2	**9.7**	14.7	56.1
Case 2	21.3	27.4	17.4	**16.1**	21.0	26.0	54.6
Case 3	**10.4**	35.0	14.6	**9.1**	30.3	12.2	56.4
Case 4	**12.6**	32.8	**12.6**	**11.6**	**11.0**	13.5	56.9
Case 5	**4.1**	15.0	18.9	**4.4**	59.1	60.0	45.8
Case 6	5.2	7.9	7.2	**1.7**	36.1	36.9	47.7
Case 7	**19.5**	22.8	29.0	**19.7**	66.4	63.1	46.5
Case 8	**3.7**	**2.7**	**4.0**	5.1	40.6	6.7	37.5
Case 9	1.5	**0.9**	10.8	**1.0**	65.4	58.5	32.5
Case 10	0.4	**0.1**	2.8	3.4	60.7	7.2	36.5

Table 5: Average Error Rate for 10 Features

	BDA7	QB	RDA	EDDA	NM	LDA	QDA
Case 1	27.9	33.3	25.4	**21.7**	**21.5**	67.0	67.0
Case 2	26.8	42.6	15.4	**12.5**	32.0	66.8	66.8
Case 3	27.2	55.7	44.8	**21.2**	47.3	66.6	66.6
Case 4	22.5	30.9	17.9	17.0	**15.5**	67.0	67.0
Case 5	1.2	30.6	11.8	**0.0**	52.4	66.6	66.6
Case 6	0.5	26.5	8.0	**0.0**	43.2	66.6	66.6
Case 7	34.7	**30.9**	41.9	63.9	66.5	66.0	66.0
Case 8	9.2	**3.5**	**2.8**	25.5	40.9	66.6	66.6
Case 9	1.3	**0.9**	62.7	32.5	66.3	66.5	66.5
Case 10	1.7	**0.9**	60.6	32.4	66.2	66.2	66.2

Table 6: Average Error Rate for 50 Features

	BDA7	QB	RDA	EDDA	NM	LDA	QDA
Case 1	35.8	31.1	28.1	**24.8**	**24.7**	68.0	68.0
Case 2	20.8	41.9	11.1	**9.0**	37.2	66.8	66.8
Case 3	46.9	56.4	50.5	**27.7**	51.1	64.9	64.9
Case 4	37.6	32.1	23.9	**21.1**	**20.3**	66.7	66.7
Case 5	**0.2**	38.3	14.5	**0.1**	55.8	66.6	66.6
Case 6	**0.1**	29.4	7.2	**0.0**	40.9	67.3	67.3
Case 7	40.0	**35.2**	44.3	64.8	66.6	65.8	65.8
Case 8	17.3	**8.1**	**7.4**	55.2	54.1	66.3	66.3
Case 9	**2.9**	**2.8**	67.0	67.0	66.6	66.6	66.6
Case 10	**2.2**	**2.4**	64.0	64.0	66.3	65.9	65.9

Table 7: Average Error Rate for 100 Features

The mean of class three is the same as the mean of class two except every odd numbered dimension of the mean is multiplied by -1.

Here, BDA7 rivals the performance of EDDA, which makes half the errors of RDA. In contrast, the error rate of QB is twice as high as BDA7 for 10 and 50 dimensions.

Case 4 is that the class means are concentrated in a high-variance subspace. The mean of class one is again located at the origin and the i^{th} component of the mean of class two is given by

$$\mu_{2i} = 2.5\sqrt{\frac{e_i}{d}\frac{i-1}{(\frac{d}{2}-1)}}, \quad 1 \le i \le d.$$

The mean of class three is the same as the mean of class two except every odd numbered dimension of the mean is multiplied by -1.

8.1.4 CASES 5 AND 6: UNEQUAL HIGHLY ELLIPSOIDAL COVARIANCE MATRICES

For these cases, the covariance matrices are highly ellipsoidal and different for each class. The eigenvalues of the class one covariance are given by Equation (26), and those of class two are given by

$$e_{2i} = \left(\frac{9(d-i)}{d-1}+1\right)^2, \quad 1 \le i \le d.$$

The eigenvalues of class three are given by

$$e_{3i} = \left(\frac{9(i-\frac{d-1}{2})}{d-1}\right)^2, \quad 1 \le i \le d.$$

For Case 5, the class means are identical. For Case 6 the class means are different, with the class one mean located at the origin and the i^{th} component of the class two mean given by $\mu_{2i} = \frac{14}{\sqrt{d}}$. The mean of class three is the same as the mean of class two except every odd numbered dimension of the mean is multiplied by -1.

In both these cases the BDA7 error falls to zero as the number of feature dimensions rise, whereas RDA plateaus around 10% error, and QB has substantially higher error. Case 5 and Case 6 present more information to the classifiers than the previous cases because the covariance matrices are substantially different. BDA7 and EDDA are able to use this information effectively to discriminate the classes.

8.1.5 CASES 7 AND 8: UNEQUAL FULL RANDOM COVARIANCES

Let R_1 be a $d \times d$ matrix where each element is drawn independently and identically from a uniform distribution on $[0,1]$. Then let the class one covariance matrix be $R_1^T R_1$. Similarly, let the class two and class three covariance matrices be $R_2^T R_2$ and $R_3^T R_3$, where R_2 and R_3 are constructed in the same manner as R_1. For Case 7, the class means are identical. For Case 8, the class means are each drawn randomly, where each element of each mean vector is drawn independently and identically from a standard normal distribution.

Case 7 is a difficult case because the means do not provide any information and the covariances may not be very different. However, BDA7 and QB only lose classification performance slowly as the dimension goes up, while EDDA jumps from 20% error at 10 feature dimensions to 64% error at 50 dimensions. For most runs of this simulation, the best EDDA model is the full covariance model, but because EDDA uses ML estimation, its estimation of the full covariance model is ill-conditioned.

Case 8 provides more information to discriminate the classes because of the different class means. EDDA again does relatively poorly because its best-choice model is the full-covariance which it estimates with ML. QB and RDA do roughly equally as well, with BDA7 at roughly twice their error.

8.1.6 CASES 9 AND 10: UNEQUAL FULL HIGHLY ELLIPSOIDAL RANDOM COVARIANCE

Let R_1, R_2, R_3 be as described for Cases 7 and 8. Then the Cases 9 and 10 covariance matrices are $R_i R_i^T R_i^T R_i$ for $i = 1,2,3$. These covariance matrices are highly ellipsoidal, often with one strong eigenvalue and many relatively small eigenvalues. This simulates the practical classification scenario in which the features are all highly correlated. For Case 9, the class means are the same. For Case 10, the class means are each drawn randomly, where each element of the mean vector is drawn independently and identically from a standard normal distribution. The two Bayesian methods capture the information and achieve very low error rates. The other classifiers do not model this situation well, and have high error rates for 50 and 100 feature dimensions.

9. Conclusions

In this paper, we have shown how a distribution-based formulation of the Bayesian quadratic discriminant analysis classifier relates to the standard parameter-based formulation, established an analytic link between Bayesian discriminant analysis and regularized discriminant analysis, and presented a functional equivalence between minimizing the expected misclassification cost and minimizing the expected Bregman divergence of class conditional distributions. A side result was the establishment of a functional definition of the standard vector Bregman divergence.

The practical contribution of this paper is the classifier BDA7, which has been shown to perform generally better than RDA, EDDA and QB on ten benchmark data sets. Key aspects of BDA7 are that the seed matrix in the inverse Wishart prior uses a coarse estimate of the covariance matrix to peg the maximum of the prior to a relevant part of the distribution-space.

The simulations presented are helpful in analyzing the different classifiers. Comparisons on simulations show that RDA and EDDA perform well when the true Gaussian distribution matches one of their regularization covariance models (e.g., diagonal, identity), but can fail when the generating distribution has a full covariance matrix, particularly when features are correlated. In contrast, the Bayesian methods BDA7 and QB can learn from the rich differentiating information offered by full covariance matrices.

We hypothesize that better priors exist, and that such priors will also be data-dependent and make use of a coarse estimate of the covariance matrix for the prior. Although modeling each class by one Gaussian distribution has too much model bias to be a general purpose classifier, Gaussian mixture model classifiers are known to work well for a variety of problems. It is an open question as to how to effectively integrate the presented ideas into a mixture model classifier.

Acknowledgments

This work was funded in part by the United States Office of Naval Research. We thank Inderjit Dhillon and Richard Olshen for helpful discussions.

Appendix A.

This appendix contains the proofs of Theorem 1 and Theorem 2.

A.1 Proof of Theorem 1

The proof employs the following identities (Box and Tiao, 1973; Gupta and Nagar, 2000),

$$\int_{\mu} \exp\left[-\frac{n_h}{2}\operatorname{tr}\left(\Sigma^{-1}(\mu-\bar{x})(\mu-\bar{x})^T\right)\right]d\mu = \left(\frac{2\pi}{n_h}\right)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}, \tag{27}$$

$$\int_{\Sigma>0}\frac{1}{|\Sigma|^{\frac{q}{2}}}\exp[-\operatorname{tr}\left(\Sigma^{-1}B\right)]d\Sigma = \frac{\Gamma_d\left(\frac{q-d-1}{2}\right)}{|B|^{\frac{q-d-1}{2}}}, \tag{28}$$

where $\Gamma_d(\cdot)$ is the multivariate gamma function,

$$\Gamma_d(a) = \left(\Gamma\left(\frac{1}{2}\right)\right)^{\frac{d(d-1)}{d}}\prod_{i=1}^{d}\Gamma\left(a-\frac{d-i}{2}\right). \tag{29}$$

To expand (5), we first simplify (7). The posterior (8) requires calculation of the normalization constant α_h,

$$\alpha_h = \int_{\Sigma_h}\int_{\mu_h}\ell(\mathcal{N}_h,\mathcal{T}_h)p(\mathcal{N}_h)\frac{d\mu d\Sigma_h}{|\Sigma_h|^{\frac{d+2}{2}}}.$$

Substitute $\ell(\mathcal{N}_h, \mathcal{T}_h)$ from (9) and $p(\mathcal{N}_h)$ from (10),

$$\alpha_h = \int_{\Sigma_h}\int_{\mu_h} \frac{\exp[-\frac{n_h}{2}\operatorname{tr}\left(\Sigma_h^{-1}(\mu_h-\bar{x}_h)(\mu_h-\bar{x}_h)^T\right)]}{(2\pi)^{\frac{n_h d}{2}}|\Sigma_h|^{\frac{n_h+q}{2}}} \exp\left[-\frac{1}{2}\operatorname{tr}\left(\Sigma^{-1}(S_h+B_h)\right)\right]\frac{d\Sigma_h d\mu_h}{|\Sigma_h|^{\frac{d+2}{2}}}.$$

Integrate with respect to μ_h using identity (27):

$$\alpha_h = \frac{1}{(2\pi)^{\frac{n_h d}{2}}}\left(\frac{2\pi}{n_h}\right)^{\frac{d}{2}}\int_{\Sigma_h}\frac{\exp[-\frac{1}{2}\operatorname{tr}\left(\Sigma_h^{-1}(S_h+B_h)\right)]}{|\Sigma_h|^{\frac{n_h+q+d+1}{2}}}d\Sigma_h.$$

Next, integrate with respect to Σ_h using identity (28):

$$\alpha_h = \frac{1}{(2\pi)^{\frac{n_h d}{2}}}\left(\frac{2\pi}{n_h}\right)^{\frac{d}{2}}\frac{\Gamma_d\left(\frac{n_h+q}{2}\right)}{\left|\frac{S_h+B_h}{2}\right|^{\frac{n_h+q}{2}}}. \tag{30}$$

Therefore the expectation $E_{N_h}[N_h(x)]$ is

$$\begin{aligned}E_{N_h}[N_h(x)] &= \int_M \mathcal{N}_h(x) f(\mathcal{N}_h) dM_h \\ &= \frac{1}{\alpha_h(2\pi)^{\frac{n_h d}{2}}}\int_{\Sigma_h}\int_{\mu_h}\frac{\exp\left[-\frac{1}{2}\operatorname{tr}\left(\Sigma_h^{-1}(x-\mu_h)(x-\mu_h)^T\right)\right]}{(2\pi)^{\frac{d}{2}}|\Sigma_h|^{\frac{1}{2}}} \\ &\cdot \frac{\exp\left[-\frac{1}{2}\operatorname{tr}\left(\Sigma_h^{-1}(S_h+B_h)\right)\right]}{|\Sigma_h|^{\frac{n_h+q}{2}}}\exp\left[-\frac{n_h}{2}\operatorname{tr}\left(\Sigma_h^{-1}(\mu_h-\bar{x}_h)(\mu_h-\bar{x}_h)^T\right)\right]\frac{d\mu_h d\Sigma_h}{|\Sigma_h|^{\frac{d+2}{2}}}.\end{aligned}$$

Integrate with respect to μ_h and Σ_h using identities (27) and (28), and equation (13) to yield

$$E_{N_h}[N_h(x)] = \frac{1}{\alpha_h(2\pi)^{\frac{dn_h}{2}}(n_h+1)^{\frac{d}{2}}}\frac{\Gamma_d\left(\frac{n_h+q+1}{2}\right)}{|A_h|^{\frac{n_h+q+1}{2}}},$$

where A_h is given by Equation (13). After substituting the value of α_h from (30),

$$E_{N_h}[N_h(x)] = \frac{1}{(2\pi)^{\frac{d}{2}}}\frac{n_h^{\frac{d}{2}}\Gamma_d\left(\frac{n_h+q+1}{2}\right)\left|\frac{S_h+B_h}{2}\right|^{\frac{n_h+q}{2}}}{(n_h+1)^{\frac{d}{2}}\Gamma_d\left(\frac{n_h+q}{2}\right)|A_h|^{\frac{n_h+q+1}{2}}}.$$

Simplify the multivariate gamma function Γ_d using (29):

$$E_{N_h}[N_h(x)] = \frac{1}{(2\pi)^{\frac{d}{2}}}\frac{n_h^{\frac{d}{2}}\Gamma\left(\frac{n_h+q+1}{2}\right)}{(n_h+1)^{\frac{d}{2}}\Gamma\left(\frac{n_h+q-d+1}{2}\right)}\frac{\left|\frac{S_h+B_h}{2}\right|^{\frac{n_h+q}{2}}}{|A_h|^{\frac{n_h+q+1}{2}}}. \tag{31}$$

Substituting (31) into (5) proves the theorem. Because $q \geq d$, the result (31) is valid for any $n_h > 0$ and any feature space dimension d.

$\square$

A.2 Proof of Theorem 2

Let

$$\begin{aligned} J[f] &= E_{N_h}[d_\phi(N_h, f)] \\ &= \int_M d_\phi(\mathcal{N}_h, f) r(\mathcal{N}_h) dM \\ &= \int_M (\phi[\mathcal{N}_h] - \phi[f] - \delta\phi[f; \mathcal{N}_h - f]) r(\mathcal{N}_h) dM, \end{aligned} \quad (32)$$

where (32) follows by substituting the definition of functional Bregman divergence (22). Consider the increment

$$\begin{aligned} \triangle J[f,a] &= J[f+a] - J[f] \quad (33) \\ &= -\int_M (\phi[f+a] - \phi[f]) r(\mathcal{N}_h) dM \\ &\quad -\int_M (\delta\phi[f+a; \mathcal{N}_h - f - a] - \delta\phi[f; \mathcal{N}_h - f]) r(\mathcal{N}_h) dM, \end{aligned} \quad (34)$$

where (34) follows from substituting (32) into (33). Using the definition of the differential of a functional (46), the first integrand in (34) can be written as

$$\phi[f+a] - \phi[f] \quad = \quad \delta\phi[f;a] + \varepsilon[f,a] \, \|a\|_{L^1(\nu)} . \quad (35)$$

Take the second integrand of (34), and subtract and add $\delta\phi[f; \mathcal{N}_h - f - a]$,

$$\begin{aligned} &\delta\phi[f+a; \mathcal{N}_h - f - a] - \delta\phi[f; \mathcal{N}_h - f] \\ &= \delta\phi[f+a; \mathcal{N}_h - f - a] - \delta\phi[f; \mathcal{N}_h - f - a] + \delta\phi[f; \mathcal{N}_h - f - a] - \delta\phi[f; \mathcal{N}_h - f] \\ &\overset{(a)}{=} \delta^2\phi[f; \mathcal{N}_h - f - a, a] + \varepsilon[f,a] \, \|a\|_{L^1(\nu)} + \delta\phi[f; \mathcal{N}_h - f] - \delta\phi[f;a] - \delta\phi[f; \mathcal{N}_h - f] \\ &\overset{(b)}{=} \delta^2\phi[f; \mathcal{N}_h - f, a] - \delta^2\phi[f; a, a] + \varepsilon[f,a] \, \|a\|_{L^1(\nu)} - \delta\phi[f;a], \end{aligned} \quad (36)$$

where (a) follows from the definition of the second differential (see (47) in Appendix B) and from the fact that

$$\delta\phi[f+a;a] - \delta\phi[f;a] = \delta^2\phi[f;a,a] + o(\|a\|^2_{L^1(\nu)}), \quad (37)$$

where $o(\|a\|^2_{L^1(\nu)})$ denotes a function which goes to 0 even if it is divided by $\|a\|^2_{L^1(\nu)}$; and (b) follows from linearity of the first term. Substitute (35) and (36) into (34),

$$\triangle J[f,a] \quad = \quad -\int_M \left(\delta^2\phi[f; \mathcal{N}_h - f, a] - \delta^2\phi[f;a,a] + \varepsilon[f,a] \, \|a\|_{L^1(\nu)} \right) r(\mathcal{N}_h) dM.$$

Note that the term $\delta^2\phi[f;a,a]$ is of order $\|a\|^2_{L^1(\nu)}$, that is, $\left\|\delta^2\phi[f;a,a]\right\|_{L^1(\nu)} \leq m \, \|a\|^2_{L^1(\nu)}$ for some constant m. Therefore,

$$\lim_{\|a\|_{L^1(\nu)} \to 0} \frac{\|J[f+a] - J[f] - \delta J[f;a]\|_{L^1(\nu)}}{\|a\|_{L^1(\nu)}} = 0,$$

where

$$\delta J[f;a] \;=\; -\int_M \delta^2\phi[f;\mathcal{N}_h - f, a] r(\mathcal{N}_h) dM. \tag{38}$$

For fixed a, $\delta^2\phi[f;\cdot,a]$ is a bounded linear functional in the second argument, so the integration and the functional can be interchanged in (38). Thus the above equation becomes

$$\delta J[f;a] = -\delta^2\phi\left[f;\int_M (\mathcal{N}_h - f)\, r(\mathcal{N}_h) dM, a\right].$$

Here we tacitly assumed that both $\int_M r(\mathcal{N}_h)dM$ and $\int_M \mathcal{N}_h\, r(\mathcal{N}_h)dM$ exist. Under the functional optimality conditions (stated in Appendix B), $J[f]$ has an extremum for $f = \hat{f}$ if, for all admissible $a \in \mathcal{A}_1$,

$$\delta^2\phi\left[f;\int_M \left(\mathcal{N}_h - \hat{f}\right) r(\mathcal{N}_h) dM, a\right] = 0. \tag{39}$$

Set $a = \int_M \left(\mathcal{N}_h - \hat{f}\right) r(\mathcal{N}_h) dM$ in (39) and use the assumption that the functional $\delta^2\phi[f;\cdot,\cdot]$ is strongly positive, which implies that the above functional can be zero if and only if $a = 0$, that is,

$$0 \;=\; \int_M (\mathcal{N}_h - \hat{f}) r(\mathcal{N}_h) dM, \tag{40}$$

$$\hat{f} \;=\; E_{N_h}[N_h]. \tag{41}$$

Equation (41) is well-defined with the measure given in (6). Because a Bregman divergence is not necessarily convex in its second argument, it is not yet established that the above unique extremum is a minimum. To show that (41) is in fact a minimum of J, from the functional optimality conditions it is enough to show that $\delta^2 J[\hat{f};\cdot,\cdot]$ is strongly positive. To show this, for $b \in \mathcal{A}_1$, consider $\delta J[f+b;a] - \delta J[f;a]$ to be the increment in the first differential of J at f. Then

$$\begin{aligned}
&\delta J[f+b;a] - \delta J[f;a] \\
&\overset{(c)}{=} \; -\int_M \left(\delta^2\phi[f+b;\mathcal{N}_h - f - b, a] - \delta^2\phi[f;\mathcal{N}_h - f, a]\right) r(\mathcal{N}_h) dM \\
&\overset{(d)}{=} \; -\int_M (\delta^2\phi[f+b;\mathcal{N}_h - f - b, a] - \delta^2\phi[f;\mathcal{N}_h - f - b, a] + \delta^2\phi[f;\mathcal{N}_h - f - b, a] \\
&\qquad -\delta^2\phi[f;\mathcal{N}_h - f, a]) r(\mathcal{N}_h) dM \\
&\overset{(e)}{=} \; -\int_M (\delta^3\phi[f;\mathcal{N}_h - f - b, a, b] + \varepsilon[f,a,a,b]\, \|b\|_{L^1(\nu)} + \delta^2\phi[f;\mathcal{N}_h - f, a] - \delta^2\phi[f;b,a] \\
&\qquad -\delta^2\phi[f;\mathcal{N}_h - f, a]) r(\mathcal{N}_h) dM \\
&\overset{(f)}{=} \; -\int_M (\delta^3\phi[f;\mathcal{N}_h - f, a, b] - \delta^3\phi[f;b,a,b] + \varepsilon[f,a,a,b]\, \|b\|_{L^1(\nu)} \\
&\qquad -\delta^2\phi[f;b,a]) r(\mathcal{N}_h) dM,
\end{aligned} \tag{42}$$

where (c) follows from using integral (38); (d) from subtracting and adding $\delta^2\phi[f;\mathcal{N}_h - f - b, a]$; (e) from using the definition of the second differential and from the fact (37); and (f) follows from the linearity of the first term and cancelation of the third and fifth terms. Note that in (42), for fixed

a, the term $\delta^3\phi[f;b,a,b]$ is of order $\|b\|^2_{L^1(\nu)}$, while the first and the last terms are of order $\|b\|_{L^1(\nu)}$. Therefore,

$$\lim_{\|b\|_{L^1(\nu)}\to 0} \frac{\left\|\delta J[f+b;a]-\delta J[f;a]-\delta^2 J[f;a,b]\right\|_{L^1(\nu)}}{\|b\|_{L^1(\nu)}}=0,$$

where

$$\delta^2 J[f;a,b] = -\int_M (\delta^3\phi[f;\mathcal{N}_b - f,a,b]-\delta^2\phi[f;b,a])r(\mathcal{N}_b)dM. \tag{43}$$

Substitute $b=a$ and $f=\hat{f}$, and interchange integration and the functional $\delta^3\phi$ in the first integral of (43) to yield

$$\begin{aligned}
\delta^2 J[\hat{f};a,a] &= -\delta^3\phi\left[\hat{f};\int_M(\mathcal{N}_b-\hat{f})r(\mathcal{N}_b)dM,a,a\right]+\int_M \delta^2\phi[\hat{f};a,a]r(\mathcal{N}_b)dM \\
&= \int_M \delta^2\phi[\hat{f};a,a]r(\mathcal{N}_b)dM \qquad (44)\\
&\geq \int_M k\|a\|^2_{L^1(\nu)} r(\mathcal{N}_b)dM \\
&= K\|a\|^2_{L^1(\nu)} > 0, \qquad (45)
\end{aligned}$$

where (44) follows from (40), and (45) follows from the strong positivity of $\delta^2\phi[\hat{f};\cdot,\cdot]$ and $K \overset{\triangle}{=} k\int_M dM > 0$. Therefore from (45) and the functional optimality conditions, $\hat{f}$ is the minimum. The result $\hat{f}$ is given in (31).

□

Appendix B. Relevant Definitions and Results from Functional Analysis

The following survey of the relevant definitions and results from functional analysis is based on Gelfand and Fomin (2000). Let a function space $\mathcal{A}_1$ be defined

$$\mathcal{A}_1 = \left\{ a:\mathbb{R}^d \to [0,1] \;\middle|\; a\in L^1(\nu),\ a>0,\ \|a\|_{L^1(\nu)}=1 \right\}$$

where ν is some measure. Note that $\mathcal{A}_1$ is a convex subspace of $L^1(\nu)$: if $a_1,a_2\in\mathcal{A}_1$ and $0\leq\omega\leq 1$, then $\omega a_1+(1-\omega)a_2\in\mathcal{A}_1$.

B.1 Definition of Continuous Linear Functionals

The functional $\psi: L^1(\nu)\to\mathbb{R}$ is linear and continuous if

1. $\psi[\omega a_1 + a_2] = \omega\psi[a_1]+\psi[a_2]$ for any $a_1,a_2\in L^1(\nu)$ and any real number ω (linearity); and

2. there is a constant C such that $|\psi[a]|\leq C\|a\|$ for all $a\in L^1(\nu)$.

B.2 Functional Derivatives

1. Let ϕ be a real functional over the normed space $L^1(\nu)$. The bounded linear functional $\delta\phi[f;\cdot]$ is the Fréchet derivative of ϕ at $f \in L^1(\nu)$ if

$$\begin{aligned}\phi[f+a]-\phi[f] &= \triangle\phi[f;a]\\ &= \delta\phi[f;a]+\epsilon[f,a]\,\|a\|_{L^1(\nu)}\end{aligned} \tag{46}$$

   for all $a \in L^1(\nu)$, with $\epsilon[f,a] \to 0$ as $\|a\|_{L^1(\nu)} \to 0$.

2. When the second variation $\delta^2\phi$ and the third variation $\delta^3\phi$ exist, they are described by

$$\begin{aligned}\triangle\phi[f;a] &= \delta\phi[f;a]+\frac{1}{2}\delta^2\phi[f;a,a]+\epsilon[f,a]\,\|a\|^2_{L^1(\nu)}\\ &= \delta\phi[f;a]+\frac{1}{2}\delta^2\phi[f;a,a]+\frac{1}{6}\delta^3\phi[f;a,a,a]+\epsilon[f,a]\,\|a\|^3_{L^1(\nu)},\end{aligned}$$

   where $\epsilon[f,a] \to 0$ as $\|a\|_{L^1(\nu)} \to 0$. The term $\delta^2\phi[f;a,b]$ is bilinear with respect to arguments a and b, and $\delta^3\phi[f;a,b,c]$ is trilinear with respect to a,b, and c.

3. Suppose $\{a_n\},\{f_n\} \subset L^1(\nu)$, moreover $a_n \to a$, $f_n \to f$,where $a,f \in L^1(\nu)$. If $\phi \in \mathcal{C}^3(L^1(\nu);\mathbb{R})$ and $\delta\phi[f;a]$, $\delta^2\phi[f;a,a]$, and $\delta^3[f;a,a,a]$ are defined as above, then $\delta\phi[f_n;a_n] \to \delta\phi[f;a]$, $\delta^2\phi[f_n;a_n,a_n] \to \delta^2\phi[f;a,a]$, and $\delta^3\phi[f_n;a_n,a_n,a_n] \to \delta^3\phi[f;a,a,a]$, respectively.

4. The functional $\delta^2\phi[f;a,a]$ defined on normed linear space $L^1(\nu)$ is quadratic; it is also ***strongly positive*** if there exists a constant $k>0$ such that $\delta^2\phi[f;a,a] \geq k\|a\|^2_{L^1(\nu)}$ for all $a \in \mathcal{A}_1$. In a finite-dimensional space, strong positivity of a quadratic form is equivalent to the quadratic form being positive definite.

B.3 Functional Optimality Conditions

A necessary condition for the differential functional $J[f]$ to have an extremum (minimum) at $f=\hat{f}$ is that $\delta J[f;a]=0$ and $\delta^2 J[f;a,a] \geq 0$, for $f=\hat{f}$ and all admissible functions $a \in \mathcal{A}_1$. A sufficient condition for a functional $J[f]$ to have a minimum for $f=\hat{f}$ is that the first variation $\delta J[f;a]$ vanishes for $f=\hat{f}$, and its second variation $\delta^2 J[f;a,a]$ is strongly positive for $f=\hat{f}$.

References

S. Amari and H. Nagaoka. *Methods of Information Geometry*. American Mathematical Society, USA, 2000.

T. W. Anderson. *An Introduction to Multivariate Statistical Analysis*. Wiley-Interscience, USA, 2003.

A. Banerjee, X. Guo, and H. Wang. On the optimality of conditional expectation as a Bregman predictor. *IEEE Trans. on Information Theory*, 51(7):2664–2669, 2005a.

A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005b.

H. Bensmail and G. Celeux. Regularized Gaussian discriminant analysis through eigenvalue decomposition. *Journal of the American Statistical Association*, 91:1743–1748, 1996.

P. J. Bickel and B. Li. Regularization in statistics. *Test*, 15(2):271–344, 2006.

G. E. P. Box and G. C. Tiao. *Bayesian Inference in Statistical Analysis*. Addison-Wesley, Reading, Massachusetts, 1973.

P. J. Brown, T. Fearn, and M. S. Haque. Discrimination with many variables. *Journal of the American Statistical Association*, 94(448):1320–1329, 1999.

S. Censor and Y. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, Oxford, England, 1997.

I. Csiszár. Generalized projections for non-negative functions. *Acta Mathematica Hungarica*, 68: 161–185, 1995.

P. S. Dwyer. Some applications of matrix derivatives in multivariate analysis. *Journal of the American Statistical Association*, 333:607–625, 1967.

B. Efron and C. Morris. Multivariate empirical Bayes and estimation of covariance matrices. *The Annals of Statistics*, 4:22–32, 1976.

J. H. Friedman. Regularized discriminant analysis. *Journal of the American Statistical Association*, 84(405):165–175, 1989.

S. Geisser. Posterior odds for multivariate normal distributions. *Journal of the Royal Society Series B Methodological*, 26:69–76, 1964.

S. Geisser. *Predictive Inference: An Introduction*. Chapman & Hall, New York, 1993.

I. M. Gelfand and S. V. Fomin. *Calculus of Variations*. Dover, USA, 2000.

A. K. Gupta and D. K. Nagar. *Matrix Variate Distributions*. Chapman and Hall/CRC, Florida, 2000.

L. R. Haff. Empirical Bayes estimation of the multivariate normal covariance matrix. *The Annals of Statistics*, 8(3):586–597, 1980.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, New York, 2001.

J. P. Hoffbeck and D. A. Landgrebe. Covariance matrix estimation and classification with limited training data. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18:763–767, 1996.

E. T. Jaynes and G. T. Bretthorst. *Probability Theory: the Logic of Science*. Cambridge University Press, Cambridge, 2003.

R. E. Kass. The geometry of asymptotic inference. *Statistical Science*, 4(3):188–234, 1989.

D. G. Keehn. A note on learning for Gaussian properties. *IEEE Trans. on Information Theory*, 11: 126–132, 1965.

E. L. Lehmann and G. Casella. *Theory of Point Estimation*. Springer, New York, 1998.

S. J. Raudys and A. K. Jain. Small sample size effects in statistical pattern recognition: Recommendations for practitioners. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(3): 252–264, 1991.

B. Ripley. *Pattern Recognition and Neural Nets*. Cambridge University Press, Cambridge, 2001.

S. Srivastava and M. R. Gupta. Distribution-based Bayesian minimum expected risk for discriminant analysis. *Proc. of the IEEE Intl. Symposium on Information Theory*, pages 2294–2298, 2006.

D. L. Swets and J. Weng. Using discriminant eigenfeatures for image retrieval. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(8):831–836, 1996.

L. Wasserman. Asymptotic inference of mixture models using data-dependent prior. *Journal of the Royal Statistical Society Series B*, 62(1):159–180, 2000.

J. Ye. Characterization of a family of algorithms for generalized discriminant analysis of undersampled problems. *Journal of Machine Learning Research*, 6:483–502, 2005.

Journal of Machine Learning Research 8 (2007) 1307-1324 Submitted 12/05; Revised 1/07; Published 6/07

From External to Internal Regret

Avrim Blum[*] AVRIM@CS.CMU.EDU
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Yishay Mansour[†] MANSOUR@CS.TAU.AC.IL
School of Computer Science
Tel Aviv University
Tel Aviv, Israel

Editor: Ron Meir

Abstract

External regret compares the performance of an online algorithm, selecting among N actions, to the performance of the best of those actions in hindsight. Internal regret compares the loss of an online algorithm to the loss of a modified online algorithm, which consistently replaces one action by another.

In this paper we give a simple generic reduction that, given an algorithm for the external regret problem, converts it to an efficient online algorithm for the internal regret problem. We provide methods that work both in the *full information* model, in which the loss of every action is observed at each time step, and the *partial information* (bandit) model, where at each time step only the loss of the selected action is observed. The importance of internal regret in game theory is due to the fact that in a general game, if each player has sublinear internal regret, then the empirical frequencies converge to a correlated equilibrium.

For external regret we also derive a quantitative regret bound for a very general setting of regret, which includes an arbitrary set of modification rules (that possibly modify the online algorithm) and an arbitrary set of time selection functions (each giving different weight to each time step). The regret for a given time selection and modification rule is the difference between the cost of the online algorithm and the cost of the modified online algorithm, where the costs are weighted by the time selection function. This can be viewed as a generalization of the previously-studied *sleeping experts* setting.

Keywords: online learning, internal regret, external regret, multi-arm bandit, sleeping experts, reductions

1. Introduction

The motivation behind regret analysis might be viewed as the following: we design a sophisticated online algorithm that deals with various issues of uncertainty and decision making, and sell it to a client. Our online algorithm runs for some time and incurs a certain loss. We would like to avoid

∗. This work was supported in part by NSF grants CCR-0105488 and IIS-0312814.
†. The work was done while the author was a fellow in the Institute of Advance studies, Hebrew University. This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778, by a grant no. 1079/04 from the Israel Science Foundation and an IBM faculty award. This publication only reflects the authors' views.

the embarrassment that our client will come back to us and claim that in *retrospect* we could have incurred a much lower loss if we used his simple alternative policy π. The regret of our online algorithm is the difference between the loss of our algorithm and the loss using π. Different notions of regret quantify differently what is considered to be a "simple" alternative policy.

At a high level one can split alternative policies into two categories. The first consists of alternative policies that are independent from the online algorithm's action selection, as is done in *external regret*. External regret, also called the *best expert* problem, compares the online algorithm's cost to the best of N actions in retrospect (see Hannan, 1957; Foster and Vohra, 1993; Littlestone and Warmuth, 1994; Freund and Schapire, 1997, 1999; Cesa-Bianchi et al., 1997). This implies that the simple alternative policy performs the same action in all time steps, which indeed is quite simple. Nonetheless, one important application of external regret is a general methodology for developing online algorithms whose performance matches that of an optimal static offline algorithm, by modeling the possible static solutions as different actions.

The second category of alternative policies are those that consider the online sequence of actions and suggest a simple modification to it, such as "every time you bought IBM, you should have bought Microsoft instead." This notion is captured by *internal regret*, introduced in Foster and Vohra (1998). Specifically, internal regret allows one to modify the online action sequence by changing every occurrence of a given action i to an alternative action j. Specific low internal regret algorithms were derived by Hart and Mas-Colell (2000), Foster and Vohra (1997, 1998, 1999), and Cesa-Bianchi and Lugosi (2003), where the use of the approachability theorem of Blackwell (1956) has played an important role in some of the algorithms.

One of the main contributions of our work is to show a simple online way to efficiently convert any low external regret algorithm into a low *internal* regret algorithm. Our guarantee is somewhat stronger than internal regret and we call it *swap regret*, which allows one to *simultaneously* swap multiple pairs of actions. (If there are N actions total, then swap-regret is bounded by N times the internal regret.) Using known results for external regret we can derive a swap regret bound of $O(\sqrt{TN\log N})$, where T is the number of time steps, which is the best known bound on swap regret for efficient algorithms. We also show an $\Omega(\sqrt{TN})$ lower bound for the case of randomized online algorithms against an adaptive adversary.

The importance of internal and swap regret is due to their tight connection to correlated equilibria, introduced by Aumann (1974). For a general-sum game of any finite number of players, a distribution Q over the joint action space is a correlated equilibrium if every player would have zero internal regret when playing it. In a repeated game scenario, if each player uses an action selection algorithm whose regret of this form is sublinear in T, then the empirical distribution of the players actions converges to a correlated equilibrium (see, e.g., Hart and Mas-Colell, 2000), and in fact, the benefit of a deviation from a correlated equilibrium is bounded exactly by R/T, where R is the largest swap regret of any of the players.

We also extend our results to the *partial information model*, also called the *adversarial multi-armed bandit* (MAB) problem in Auer et al. (2002a). In this model, the online algorithm only gets to observe the loss of the action actually selected, and does not see the losses of the actions not chosen. For example, if you are driving to work and need to select which of several routes to take, you only observe the travel time on the route actually taken. If we view this as an online problem, each day selecting which route to take on that day, then this fits the MAB setting. Furthermore, the route-choosing problem can be viewed as a general-sum game: your travel time depends on the choices of the other drivers as well. Thus, if *every* driver uses a low internal-regret algorithm, then the uniform distribution over observed traffic patterns will converge to a correlated equilibrium. For the MAB problem, our combining algorithm requires additional assumptions on the base external-regret MAB algorithm: a smoothness in behavior when the actions played are taken from a somewhat different distribution than the one proposed by the algorithm. Luckily, these conditions are satisfied by existing external-regret MAB algorithms such as that of Auer et al. (2002a). For the multi-armed bandit setting, we derive an $O(N\sqrt{NT\log N})$ swap-regret bound. Thus, after $T = O(\frac{1}{\varepsilon^2}N^3\log N)$ rounds, the empirical distribution on the history is an ε-correlated equilibrium. In a recent work, Stoltz (2005) gives an improved swap regret bound of $O(N\sqrt{T\log N})$. (The work of Hart and Mas-Colell (2001) also gives a multi-armed bandit algorithm whose swap regret is sublinear in T, but does not derive explicit bounds.)

One can also envision broader classes of regret. Lehrer (2003) defines a notion of *wide range regret* that allows for arbitrary action-modification rules, which might depend on history, and also Boolean time selection functions (that determine which subset of times is relevant). Using the approachability theorem, he shows a scheme that in the limit achieves no regret (i.e., regret is sublinear in T). While Lehrer (2003) derives the regret bounds in the limit, we derive finite-time regret bounds for this setting. We show that for any family of N actions, M time selection functions and K modification rules, the maximum regret with respect to any selection function and modification rule is bounded by $O(\sqrt{TN\log(MK)})$. Our model also handles the case where the time selection functions are not Boolean, but rather real valued in $[0,1]$.

This latter result can be viewed as a generalization of the *sleeping experts* setting of Freund et al. (1997) and Blum (1997). In the sleeping experts problem, we again have a set of experts, but on any given time step, each expert may be awake (making a prediction) or asleep (not predicting). This is a natural model for combining a collection of if-then rules that only make predictions when the "if" portion of the rule is satisfied, and this setting has had application in domains ranging from managing a calendar (Blum, 1997) and text-categorization (Cohen and Singer, 1999) to learning how to formulate web search-engine queries (Cohen and Singer, 1996). By converting each such sleeping-expert into a pair $\langle$expert, time-selection function$\rangle$, we achieve the desired guarantee that for each sleeping-expert, our loss *during the time that expert was awake* is not much more than its loss in that period. Moreover, by using non-Boolean time-selection functions, we can naturally handle prediction rules that have varying degrees of confidence in their predictions and achieve a confidence-weighted notion of regret.

We also study the case of deterministic Boolean prediction in the setting of time selection functions. We derive a deterministic online algorithm whose number of weighted errors, with respect to any time selection function from our class of M selection functions, is at most $3OPT + 2 + 2\log_2 M$, where OPT is the cumulative loss of the best constant prediction for that time selection function.

1.1 Related Work

A different conversion procedure from external to internal regret was given independently by Stoltz and Lugosi (2005), yet the approach there is very different from the one developed here. Further results regarding the relation between external and internal regret appear in Stoltz and Lugosi (2007) and for the multi-armed bandit setting in Cesa-Bianchi et al. (2006). In comparison to Stoltz and Lugosi (2007), we are able to achieve a better swap regret guarantee in polynomial time. (A straightforward application of Stoltz and Lugosi (2007) to swap regret would require time-complexity $\Omega(N^N)$; alternatively, they can achieve a good internal-regret bound in polynomial time, but then their swap regret bound becomes worse by a factor of $\sqrt{N}$.) On the other hand, their work is applicable to any convex loss function while our work is restricted to linear loss functions. See also Stoltz (2005), Section 1.3, for a discussion of the different procedures.

2. Model and Preliminaries

We assume an adversarial online model where there are N available actions $\{1,\ldots,N\}$. At each time step t, an online algorithm H selects a distribution p^t over the N actions. After that, the adversary selects a loss vector $\ell^t \in [0,1]^N$, where $\ell_i^t \in [0,1]$ is the loss of the i-th action at time t. In the *full information model*, the online algorithm receives the loss vector ℓ^t and experiences a loss $\ell_H^t = \sum_{i=1}^N p_i^t \ell_i^t$. In the *partial information model*, the online algorithm receives $(\ell_{k^t}^t, k^t)$, where k^t is distributed according to p^t, and $\ell_H^t = \ell_{k^t}^t$ is its loss. The loss of the i-th action during the first T time steps is $L_i^T = \sum_{t=1}^T \ell_i^t$, and the loss of H is $L_H^T = \sum_{t=1}^T \ell_H^t$. The aim for the external regret setting is to design an online algorithm that will be able to approach the best action, namely, to have a loss close to $L_{min}^T = \min_i L_i^T$. Formally we would like to minimize the external regret $R = L_H^T - L_{min}^T$.

We introduce a notion of a *time selection* function. A time selection function I is a function over the time steps mapping each time step to $[0,1]$. That is, $I : \{1,\ldots,T\} \to [0,1]$. The loss of action j using time-selector I is $L_{j,I}^T = \sum_t I(t)\ell_j^t$. Similarly we define $L_{H,I}$, the loss of the online algorithm H with respect to time selection function I, as $L_{H,I}^T = \sum_t I(t)\ell_H^t$, where ℓ_H^t is the loss of H at time t.[1] This notion of experts with time selection is very similar to the notion of "sleeping experts" studied in Freund et al. (1997). Specifically, for each action j and time selection function I, one can view the pair (j,I) as an expert that is "awake" when $I(t) = 1$ and "asleep" when $I(t) = 0$ (and we could view it as "partially awake" when $I(t) \in (0,1)$).

We also consider modification rules that modify the actions selected by the online algorithm, producing an alternative strategy we will want to compete against. A *modification rule* F has as input the history and an action choice and outputs a (possibly different) action. (We denote by F^t the function F at time t, including any dependency on the history.) Given a sequence of probability distributions p^t used by an online algorithm H, and a modification rule F, we define a new sequence of probability distributions $f^t = F^t(p^t)$, where $f_i^t = \sum_{j:F^t(j)=i} p_j^t$. The loss of the modified sequence is $L_{H,F} = \sum_t \sum_i f_i^t \ell_i^t$. Similarly, given a time selection function I and a modification rule F we define $L_{H,I,F} = \sum_t \sum_i I(t) f_i^t \ell_i^t$.

In our setting we assume a finite class of N actions, $\{1,\ldots,N\}$, a finite set $\mathcal{F}$ of K modification rules, and a finite set I of M time selection function. Given a sequence of loss vectors, the regret of an online algorithm H with respect to the N actions, the K modification rules, and the M time selection functions, is

$$R_H^{I,\mathcal{F}} = \max_{I\in I} \max_{F\in\mathcal{F}} \{L_{H,I} - L_{H,I,F}\}.$$

Note that the external regret setting is equivalent to having a single time-selection function ($I(t) = 1$ for all t) and a set $\mathcal{F}^{ex}$ of N modification rules F_i, where F_i always outputs action i. For internal regret, the set $\mathcal{F}^{in}$ consists of $N(N-1)$ modification rules $F_{i,j}$, where $F_{i,j}(i) = j$ and

1. We can let the time selector depend on the history up to time t, rather than the time t itself, and all the results presented would be the same.

$F_{i,j}(i') = i'$ for $i' \neq i$, plus the identity function. That is, the internal regret of H is

$$\max_{F \in \mathcal{F}^{in}} \{L_H - L_{H,F}\} = \max_{i,j} \sum_t p_i^t (\ell_i^t - \ell_j^t).$$

We also define an extension to internal regret that we call *swap regret*. This case has $\mathcal{F}^{sw}$ include all N^N functions $F : \{1, \ldots, N\} \to \{1, \ldots, N\}$, where the function F swaps the current online action i with $F(i)$ (which can be the same or a different action).[2]

A few simple relationships between the different types of regret: since $\mathcal{F}^{ex} \subseteq \mathcal{F}^{sw}$ and $\mathcal{F}^{in} \subseteq \mathcal{F}^{sw}$, both external and internal regret are upper-bounded by swap-regret. Also, swap-regret is at most N times larger than internal regret. On the other hand, even with $N = 3$, there are simple examples that separate internal and external regret (see, e.g., Stoltz and Lugosi, 2005).

2.1 Correlated Equilibria and Swap Regret

We briefly sketch the relationship between correlated equilibria and swap regret.

Definition 1 *A game $G = \langle M, (A_i), (s_i) \rangle$ has a finite set M of m players. Player i has a set A_i of N actions and a loss function $s_i : A_i \times (\times_{j \neq i} A_j) \to [0,1]$ that maps the action of player i and the actions of the other players to a real number. (We have scaled losses to $[0,1]$.)*

The aim of each player is to minimize its loss. A correlated equilibrium is a distribution P over the joint action space with the following property. Imagine a correlating device draws a vector of actions $\vec{a}$ using distribution P over $\times A_i$, and gives player i the action a_i from $\vec{a}$. (Player i is not given any other information regarding $\vec{a}$.) The probability distribution P is a correlated equilibrium if, for each player, it is its best response to play the suggested action (provided that the other players do not deviate).

We now define an ε-correlated equilibrium.

Definition 2 *A joint probability distribution P over $\times A_i$ is an ε-correlated equilibrium if for every player j and for any function $F : A_j \to A_j$, we have $E_{a \sim P}[s_j(a_j, a^{-j})] \leq E_{a \sim P}[s_j(F(a_j), a^{-j})] + \varepsilon$, where a^{-j} denotes the joint actions of the other players.*

In other words, P is an ε-correlated equilibrium if the expected incentive to deviate is at most ε for every player.

The following theorem relates the empirical distribution of the actions performed by each player, their swap regret, and the distance from a correlated equilibrium (see also, Foster and Vohra 1997, 1998 and Hart and Mas-Colell 2000).

Theorem 3 *Let $G = \langle M, (A_i), (s_i) \rangle$ be a game and assume that for T time steps each player follows a strategy that has swap regret of at most $R(T,N)$. The empirical distribution Q of the joint actions played by the players is an $(R(T,N)/T)$-correlated equilibrium, and the loss of each player equals, by definition, its expected loss on Q.*

The above states that the payoff of each player is its payoff in some approximate correlated equilibrium. In addition, it relates the swap regret to the distance from a correlated equilibrium. Note that if the average swap regret vanishes then the procedure converges, in the limit, to the set of correlated equilibria (see Hart and Mas-Colell 2000 and Foster and Vohra 1997, 1999).

2. Note that in swap and external regret, the modification functions do not depend on history. In Section 7 we consider general modification functions.

3. Generic Reduction from External to Swap Regret

We now give a black-box reduction showing how any algorithm A achieving good external regret can be used as a subroutine to achieve good swap regret as well. The high-level idea is as follows. We will instantiate N copies of the external-regret algorithm. At each time step, these algorithms will each give us a probability vector, which we will combine in a particular way to produce our own probability vector p. When we receive a loss vector ℓ, we will partition it among the N algorithms, giving algorithm A_i a fraction p_i (p_i is our probability mass on action i), so that A_i's belief about the loss of action j is $\sum_t p_i^t \ell_j^t$, and matches the cost we would incur putting i's probability mass on j. In the proof, algorithm A_i will in some sense be responsible for ensuring low regret of the $F_{i,j}$ variety. The key to making this work is that we will be able to define the p's so that the sum of the losses of the algorithms A_i on their own loss vectors matches our overall true loss.

To be specific, let us formalize what we mean by an external regret algorithm.

Definition 4 *An algorithm A has external regret $R(L_{min},T,N)$ if for any sequence of T losses ℓ^t such that some action has total loss at most L_{min}, for any action $j \in \{1,\dots,N\}$ we have*

$$L_A^T = \sum_{t=1}^{T} \ell_A^t \le \sum_{t=1}^{T} \ell_j^t + R(L_{min},T,N) = L_j^T + R(L_{min},T,N) \; .$$

We assume we have N algorithms A_i (which could all be identical or different) such that A_i has external regret $R_i(L_{min},T,N)$. We combine the N algorithms as follows. At each time step t, each algorithm A_i outputs a distribution q_i^t, where $q_{i,j}^t$ is the fraction it assigns action j. We compute a vector p such that $p_j^t = \sum_i p_i^t q_{i,j}^t$. That is, $p = pQ$, where p is the row-vector of our probabilities and Q is the matrix of $q_{i,j}$. (We can view p as a stationary distribution of the Markov Process defined by Q, and it is well known such a p exists and is efficiently computable.) For intuition into this choice of p, notice that it implies we can consider action selection in two equivalent ways. The first is simply using the distribution p to select action j with probability p_j. The second is to select algorithm A_i with probability p_i and then to use algorithm A_i to select the action (which produces distribution pQ).

When the adversary returns ℓ^t, we return to each A_i the loss vector $p_i^t \ell^t$. So, algorithm A_i experiences loss $(p_i^t \ell^t)\cdot q_i^t = p_i^t(q_i^t \cdot \ell^t)$.

Now we consider the guarantee that we have for algorithm A_i, namely, for any action j,

$$\sum_{t=1}^{T} p_i^t(q_i^t \cdot \ell^t) \quad \le \quad \sum_{t=1}^{T} p_i^t \ell_j^t + R_i(L_{min},T,N) \; . \tag{1}$$

If we sum the losses of the N algorithms at any time t, we get $\sum_i p_i^t(q_i^t \cdot \ell^t) = p^t Q^t \ell^t$, where p^t is the row-vector of our probabilities, Q^t is the matrix of $q_{i,j}^t$, and ℓ^t is viewed as a column-vector. By design of p^t, we have $p^t Q^t = p^t$. So, the sum of the perceived losses of the N algorithms is equal to our actual loss $p^t \ell^t$.

Therefore, summing Equation (1) over all N algorithms, the left-hand-side sums to L_H^T. Since the right-hand-side of Equation (1) holds for any j, we have that for any function $F : \{1,\dots,N\} \to \{1,\dots,N\}$,

$$L_H^T \le \sum_{i=1}^{N} \sum_{t=1}^{T} p_i^t \ell_{F(i)}^t + \sum_{i=1}^{N} R_i(L_{min},T,N).$$

We have therefore proven the following theorem.

Theorem 5 *For any N algorithms A_i with regret R_i, for every function $F : \{1,\dots,N\} \to \{1,\dots,N\}$, for any sequence of T losses ℓ^t such that some action has total loss at most L_{min}, the above algorithm satisfies*

$$L_H \le L_{H,F} + \sum_{i=1}^{N} R_i(L_{min}, T, N),$$

that is, the swap-regret of H is at most $\sum_{i=1}^{N} R_i(L_{min}, T, N)$.

A typical *optimized experts algorithm*, such as in Littlestone and Warmuth (1994), Freund and Schapire (1997), Auer et al. (2002a), and Cesa-Bianchi et al. (1997), will have $R(L_{min}, T, N) = O(\sqrt{L_{min}\log N} + \log N)$. (Alternatively, Corollary 15 can be also used to deduce the above bound.) We can immediately derive the following corollary.

Corollary 6 *Using an optimized experts algorithm as the A_i, for every function $F : \{1,\dots,N\} \to \{1,\dots,N\}$, we have that*

$$L_H \le L_{H,F} + O(N\sqrt{T\log N})\ .$$

We can perform a slightly more refined analysis of the bound by having L^i_{min} be the minimum loss for an action in A_i. Note that $\sum_{i=1}^{N} L^i_{min} \le T$, since we scaled the losses given to algorithm A_i at time t by p^t_i. By convexity of the square-root function, this implies that $\sum_{i=1}^{N} \sqrt{L^i_{min}} \le \sqrt{NT}$, which implies the worst case regret is $O(\sqrt{TN\log N})$.[3]

Corollary 7 *Using optimized experts algorithms as the A_i, for every function $F : \{1,\dots,N\} \to \{1,\dots,N\}$, we have that*

$$L_H \le L_{H,F} + O(\sqrt{TN\log N})\ .$$

One strength of the above general reduction is it ability to accommodate new regret minimization algorithms. For example, using the algorithms of Cesa-Bianchi et al. (2005) one can get a more refined regret bound, which depends on the second moment.

4. Lower Bounds on Swap Regret

Notice that while good algorithms for external regret achieve bounds of $O(\sqrt{T\log N})$, our swap-regret bounds are roughly $O(\sqrt{TN\log N})$. Or, to put it another way, imagine we are interested in the number of time steps T needed to achieve an average regret of ε per time step (a total regret of εT). Then, for external regret we have algorithms that can do this in $T = O(\varepsilon^{-2}\log N)$ steps, whereas our bounds require $T = O(\varepsilon^{-2}N\log N)$ steps for the case of swap-regret. From the point of view of equilibrium, this means that while for *two-player zero-sum* games such algorithms will achieve approximate minimax-optimality in $O(\log N)$ steps when played against each other, for *general-sum* games we seem to need $O(N\log N)$ steps to achieve an ε-correlated equilibrium. A natural question is whether this is best possible. In particular, is it possible to guarantee swap-regret at most εT in a number of time steps that is sublinear in the size of the action space N?

3. We need to use here an external regret algorithm which does not need to have as an input the value of L^i_{min}. An example of such an algorithm is Corollary 2 in Cesa-Bianchi et al. (2005), which guarantees an external regret of at most $O(\sqrt{L_{min}\log N} + \log N)$.

We give here a partial answer: a lower bound of $\Omega(\sqrt{TN})$ on swap-regret but in a more adversarial model. Specifically, we have defined swap regret with respect to the *distribution* p^t produced by the player, rather than the actual action a_t selected from that distribution. In the case that the adversary is oblivious (does not depend on the player's action selection) then the two models have the same expected regret. However we will consider an *adaptive* adversary, whose choices may depend on the player's action selection in previous rounds. In this setting (adaptive adversary *and* regret defined with respect to the action selected from p^t rather than p^t itself) we derive an $\Omega(\sqrt{TN})$ lower bound.

Before presenting these bounds, we first mention one subtle issue. For a given stochastic adversary, the optimal policy for minimizing *loss* may not be the optimal policy for minimizing swap-regret. For example, consider a process in which $\{0,1\}$ losses are generated by a fair coin, except that in time steps $t \in ((i-1)T/N, iT/N]$, action i has loss of 1 with probability only $1/2-1/T$. In this case, the optimal policy for minimizing expected *loss* uses action $i=1$ for the first T/N steps, then action $i=2$ for the next T/N steps, and so forth. However, because of the variance of the coin flips, in retrospect each action played can be swapped with another for an expected gain of $\Omega(\sqrt{(T\log N)/N})$ each (see, e.g., Feller, 1968, 1971), giving a total swap-regret of $\Omega(\sqrt{TN\log N})$ for this policy. On the other hand, a policy that just picks a single fixed action would have swap-regret only $O(\sqrt{T\log N})$ even though its expected loss is slightly higher.

We now present our lower bound, first giving a somewhat weaker version whose proof is simpler but contains the main ideas, and then giving the stronger version. Notice that even the weaker version (Theorem 8) implies that $\Omega(N/\varepsilon)$ time steps are necessary in order to achieve an average swap regret ε per time step, in the adaptive adversary model.

Theorem 8 *There exists an adaptive adversary such that for any randomized online algorithm A, its expected swap regret $E[\max_{F\in\mathcal{F}^{sw}} L_A - L_{A,F}]$ (defined with respect to the actions selected from p^t rather than p^t itself), is at least* $\min((N-1)/16, T/8)$.

Proof The adversary behaves as follows. At each time step t, all actions i that have been previously played by algorithm A receive a loss of 1. All other actions i receive either (a) a loss chosen uniformly and independently at random from $\{0,1\}$ if $i \in \{1,\dots,N/2\}$, or (b) a loss of exactly 1/2 if $i \in \{N/2+1,\dots,N\}$. The basic idea of the argument now is that so long as there are still actions of type (a) remaining, algorithm A has no good choices: if it chooses to play a previously-played action, then it will incur large loss, whereas if it chooses to play a new action, this will have a good probability of substantially increasing swap regret. The presence of the actions of type (b) with loss exactly $1/2$ is not really necessary but helps to simplify the calculations. Formally, we argue as follows.

Assume $T < N/2$. We will give a swap-regret lower bound of $T/8$, implying our desired result. In particular, we will keep track of two quantities: E_F, the expected regret of A with respect to a specific modification rule F, and E_L, the expected loss of A. Modification rule F is defined as follows: the first time t that algorithm A plays some action i, we define $F(i)$ to be whichever action performed best at time t (breaking ties arbitrarily); for actions never played, the value of F does not matter. Now, consider some specific time step t. If algorithm A plays some action i that was never before played, then E_F increases by at least $1/4$ (since for $T < N/2$ there must be at least one other unplayed action $j \le N/2$ and $\mathbf{E}[\max(\ell_i^t - \ell_j^t, 0)] = 1/4$), and E_L increases by $1/2$. On the other hand, if algorithm A plays a previously-played action, then E_L increases by 1, and E_F at least does not decrease. Notice that either way, $2E_F + E_L$ increases by at least 1. Therefore, by time T we have $2E_F + E_L \geq T$. Now, if algorithm A is such that $E_F \geq T/8$ then we are done (the expected regret with respect to F is large). On the other hand, if $E_F < T/8$, then this implies $E_L \geq 3T/4$. However, this means that algorithm A has large expected *external* regret since there must exist some unplayed action $j > N/2$ whose total loss is exactly $T/2$. Thus in this case the expected swap-regret of A is at least $T/4$. ■

Theorem 9 *There exists an adaptive adversary such that for any randomized online algorithm A, its expected swap regret $E[\max_{F\in\mathcal{F}^{sw}} L_A - L_{A,F}]$ (defined with respect to the actions selected from p^t rather than p^t itself), is at least $\sqrt{TN}/160 - 1$, for $N \leq T \leq \frac{1}{\sqrt{N}} e^{N/288}$.*

Proof The adversary is the same as that used in the proof of Theorem 8, except now it waits until an action is played $8T/N$ times before causing it to deterministically receive a loss of 1. Specifically, all actions played by algorithm A at least $8T/N$ times receive a loss of 1, and all other actions i receive either (a) a loss chosen uniformly and independently at random from $\{0,1\}$ if $i \in \{1,\ldots,N/2\}$, or (b) a loss of exactly 1/2 if $i \in \{N/2+1,\ldots,N\}$. Let us call an action that has been played less than $8T/N$ times a "low-loss action" and those played at least $8T/N$ times a "1-loss action". Let T^ℓ denote the number of times the algorithm plays low-loss actions (which could be a random variable depending on the algorithm). Notice that the expected loss of the algorithm is $\mathbf{E}[T^\ell/2 + (T - T^\ell)] = T - \mathbf{E}[T^\ell/2]$.

We break the argument into two cases based on $\mathbf{E}[T^\ell]$. The simpler case is $\mathbf{E}[T^\ell] \leq 3T/4$ (i.e., the algorithm plays many 1-loss actions). In that case, the expected loss of the algorithm is at least $5T/8$. On the other hand, there must be some action of total loss at most $T/2$ because it is not possible for the algorithm to have played all actions $i \in \{N/2+1,\ldots,N\}$ for $8T/N$ times each. So, the expected regret is at least $T/8 \geq \sqrt{TN}/8$.

We now analyze the case that $\mathbf{E}[T^\ell] > 3T/4$. Let $\mathcal{T}_i$ denote the time steps in which the algorithm plays i, and let $T_i = |\mathcal{T}_i|$. Define modification rule F such that $F(i)$ is the action of least total loss in time steps $\mathcal{T}_i$. We will argue later that with probability $1-\lambda$ (where we will set $\lambda = 1/T$), for all i the action $F(i)$ has loss at most $T_i/2 - \sqrt{T_i}/5$ during time steps $\mathcal{T}_i$. So, letting R denote the swap-regret of algorithm A, the expected swap-regret $\mathbf{E}[R]$ is at least the difference between its expected loss and $\mathbf{E}[\sum_i T_i/2 - \sqrt{T_i}/5] + \lambda T$:

$$\mathbf{E}[R] \quad \geq \quad T - \mathbf{E}\left[\frac{T^\ell}{2}\right] - \mathbf{E}\left[\sum_{i=1}^{N} \frac{T_i}{2} - \frac{\sqrt{T_i}}{5}\right] - \lambda T \geq \frac{1}{5}\mathbf{E}\left[\sum_{i=1}^{N} \sqrt{T_i}\right] - \lambda T,$$

where we use the fact that $\sum_i T_i = T$ and $T^\ell \leq T$.

The number of actions i such that $T_i \geq T/(4N)$ is at least $(T^\ell - T/4)/(8T/N)$, since even if one considers only the time steps in which low-loss actions are played, at most $T/4$ of them can involve playing actions with $T_i < T/(4N)$, and for the rest, any given action i can occur at most $8T/N$ times. Since $\mathbf{E}[T^\ell] \geq 3T/4$, the *expected* number of such actions is at least $(T/2)/(8T/N) = N/16$ and therefore,

$$\mathbf{E}[R] \quad \geq \quad \frac{1}{5}\mathbf{E}\left[\sum_{i=1}^{N} \sqrt{T_i}\right] - \lambda T \quad \geq \quad \frac{N}{80}\sqrt{\frac{T}{4N}} - \lambda T \quad = \quad \frac{\sqrt{TN}}{160} - \lambda T.$$

It remains to show that with high probability, all actions $F(i)$ have loss at most $T_i/2 - \sqrt{T_i}/5$ during time steps $\mathcal{T}_i$. First, note that in K coin tosses, with probability at least $1/3$ we have at most $K/2 - \sqrt{K}/5$ heads. Fix an action i and any given value of T_i. So, by Hoeffding bounds, if $N/2$ coins (corresponding to actions $1,\ldots,N/2$) are each tossed $K = T_i$ times, with probability at least $e^{-N(1/3-1/4)^2}$ we have over $N/8$ (i.e., $1/4$ of them) with at most $T_i/2 - \sqrt{T_i}/5$ heads. Thus, even if the algorithm could decide which (at most $N/8$) actions to turn into 1-loss actions after the fact, with probability at least $e^{-N(1/3-1/4)^2}$ there will still be at least one with loss at most $T_i/2 - \sqrt{T_i}/5$. Summing over all i and all possible values of T_i yields a failure probability at most $NTe^{-N(1/3-1/4)^2} = \lambda$. For $T \leq \frac{1}{\sqrt{N}}e^{N/288}$, this is at most $1/T$, completing the proof. ■

5. Reducing External to Swap Regret in the Partial Information Model

In the full information setting the learner gets, at the end of each time step, full information on the costs of all the actions. In the partial information (multi-arm bandit) model, the learner gets information only about the action that was selected. In some applications this is a more plausible model regarding the information the learner can observe.

The reduction in the partial information model is similar to the one of the full information model, but with a few additional complications. We are given N partial information algorithms A_i. At each time step t, each algorithm A_i outputs a distribution q_i^t. Our master online algorithm combines them to some distribution p^t which it uses. Given p^t it receives a feedback, but now this includes information only regarding one action, that is, it receives $(\ell_{k^t}^t, k^t)$, where k^t is distributed according to p^t. We take this feedback and distribute to each algorithm A_i a feedback (c_i^t, k^t), such that $\sum_i c_i^t = \ell_{k^t}^t$. The main technical difficulty is that now the action selected, k^t, is distributed according to p^t and not q_i^t. (For example, it might be that A_i has $q_{i,j}^t = 0$ but it receives feedback about action j. From A_i's point of view this is impossible! Or, more generally, A_i might start noticing it seems to have a very bad random-number generator.) For this reason, for the reduction to work we need to make a stronger assumption about the guarantees of the algorithms A_i, which luckily is implicit in the algorithms of Auer et al. (2002a). Since results of Auer et al. (2002a) are stated in terms of maximizing gain rather then minimizing loss we will switch to this notation, for example, define the benefit of action j at time t to be $b_j^t = 1 - \ell_j^t$.

We start by describing our MAB algorithm *SR_MAB*. Initially, we are given N partial information algorithms A_i. At each time step t, each A_i gives a *selection distribution* q_i^t over actions. Given all the selection distributions we compute an *action distribution* p^t. We would like to keep two sets of gains: one is the *real gain*, denoted by b_i^t, and the other is the gain that the MAB algorithm A_i observes, $g_{A_i}^t$. Given the action distribution p^t the adversary selects a vector of real gains b_i^t. Our MAB algorithm *SR_MAB* receives a single feedback $(b_{k^t}^t, k^t)$ where k^t is a random variable that with probability p_j^t equals j. Algorithm *SR_MAB*, given b^t, returns to each A_i a pair $(g_{A_i}^t, k^t)$, where the observed gain $g_{A_i}^t$ is based on $b_{k_t}^t$, p^t and q_i^t. Again, note that k^t is distributed according to p^t, which may not equal q_i^t: it is for this reason we need to use an MAB algorithm that satisfies certain properties (stated in Lemma 10).

In order to specify our MAB algorithm, *SR_MAB*, we need to specify how it selects the action distribution p^t and the observed gains $g_{A_i}^t$. As in the full information case, we compute an action distribution p^t such that $p_j^t = \sum_i p_i^t q_{i,j}^t$. That is, $p = pQ$, where p is the row-vector of our probabilities and Q is the matrix of $q_{i,j}$. Given p^t the adversary returns a real gain $(b^t_{k^t}, k^t)$, namely, the real gain is of our algorithm $b^t_{k^t}$. We return to each algorithm A_i an observed gain of $g^t_{A_i} = p^t_i b^t_{k^t} q_{i,k^t} / p^t_{k^t}$. (In general, define $g^t_{i,j} = p^t_i b^t_j q^t_{i,j} / p^t_j$, if $j = k^t$ and $g^t_{i,j} = 0$ if $j \neq k^t$.)

First, we will show that $\sum_{i=1}^N g^t_{A_i} = b^t_{k^t}$ which implies that $g^t_{A_i} \in [0,1]$. From the property of the distribution p^t we have that,

$$\sum_{i=1}^N g^t_{A_i} = \sum_{i=1}^N \frac{p^t_i b^t_{k^t} q_{i,k^t}}{p^t_{k^t}} = \frac{p^t_{k^t} b^t_{k^t}}{p^t_{k^t}} = b^t_{k^t}.$$

This shows that we distribute our real gain among the algorithms A_i; that is, that the sum of the observed gains equals the real gain. In addition, it bounds the observed gain that each algorithm A_i receives. Namely, $0 \le g^t_{A_i} \le b^t_{k^t} \le 1$.

In order to describe the guarantee that each external regret multi-arm bandit algorithm A_i is required to have, we need the following additional definition. At time t let $X^t_{i,j}$ be a random variable such that $X^t_{i,j} = g^t_{i,j}/q^t_{i,j}$ if $j = k^t$ and $X^t_{i,j} = 0$ otherwise. The expectation of $X^t_{i,j}$ is,

$$E_{k^t \sim p^t}[X^t_{i,k^t}] \;=\; p^t_{k^t} \frac{g^t_{i,k^t}}{q^t_{i,k^t}} \;=\; p^t_{k^t} \frac{p^t_i b^t_{k^t}}{p^t_{k^t}} \;=\; p^t_i b^t_{k^t}.$$

Lemma 10 (Auer et al. 2002a) *There exists a multi-arm bandit algorithm, A_i, such that for any sequence of observed gains $g^t_{i,j} \in [0,1]$ it outputs actions distributions q^t_i, and for any sequence of selected actions k^t, and for any action r and parameter $\gamma \in (0,1]$, then,*

$$G_{A_i,g^t} \equiv \sum_{t=1}^T g^t_{A_i} \equiv \sum_{t=1}^T g^t_{i,k^t} \ge (1-\gamma) \sum_{t=1}^T X^t_{i,r} - \frac{N \ln N}{\gamma} - \frac{\gamma}{N} \sum_{t=1}^T \sum_{j=1}^N X^t_{i,j},$$

where $X^t_{i,j}$ is a random variable such that $X^t_{i,j} = g^t_{i,j}/q^t_{i,j}$ if $j = k^t$ and $X^t_{i,j} = 0$ otherwise.

Note that in Auer et al. (2002a) the action distribution is identical to the selection distribution, that is, $p^t \equiv q^t$, and the observed and real gain are identical, that is, $g^t \equiv b^t$. Auer et al. (2002a) derive the external regret bound by taking the expectation with respect to the action distribution (which is identical to the selection distribution). In our case we separate the real gain from the observed gain, which adds another layer of complication. (Technically, the distribution p^t is a random variable that depends on the history $\mathcal{H}^{t-1}$ up to time t, that is, the observed actions $k^1, \ldots k^{t-1}$ and well as the observed gains $b^1_{k^1}, \ldots b^{t-1}_{k^{t-1}}$. For this reason we take the expectation with respect to $\mathcal{H}^{t-1}$ every time we refer to p^t. For simplicity we make this dependency implicitly in the expectations $E[\cdot]$.) We define the expected benefit of *SR_MAB* to be $B_{SR\_MAB} = E[\sum_{t=1}^T b^t_{SR\_MAB}]$ and for a function $F : \{1,\ldots,N\} \to \{1,\ldots,N\}$ we define $B_{SR\_MAB,F} = E[\sum_{t=1}^T \sum_{i=1}^N p^t_i b^t_{F(i)}]$. We now state our main theorem regarding the partial information model.

Theorem 11 *Given a multi-arm bandit algorithm satisfying Lemma 10 (such as the algorithm of Auer et al., 2002a), it can be converted to a master online algorithm SR_MAB, such that*

$$B_{SR\_MAB} \ge \max_F B_{SR\_MAB,F} - N \cdot R^{MAB}(B_{max}, T, N),$$

where the expectation is over the observed actions of SR_MAB, B_{max} bounds the maximum benefit of any algorithm and $R^{MAB}(B,T,N) = O(\sqrt{BN \log N})$.

Proof Let the total observed gain of algorithm A_i be $G_{A_i} = \sum_{t=1}^{T} g^t_{A_i} = \sum_{t=1}^{T} g^t_{i,k^t}$. Since we distribute our gain between the A_i, that is, $\sum_{i=1}^{N} g^t_{A_i} = b^t_{SR\_MAB}$, we have that $B_{SR\_MAB} = E[\sum_{t=1}^{T} b^t_{SR\_MAB}] = \sum_{i=1}^{N} E[G_{A_i}]$. Since $g^t_{i,j} \in [0,1]$, by Lemma 10, this implies that for any action r, after taking the expectation, we have

$$\begin{aligned}
E[G_{A_i}] &= E[\sum_{t=1}^{T} E_{p^t}[g^t_{i,k^t}]] \\
&\geq (1-\gamma)E[\sum_{t=1}^{T} E_{p^t}[X^t_{i,r}]] - \frac{N \ln N}{\gamma} - \frac{\gamma}{N} E\left[\sum_{t=1}^{T}\sum_{j=1}^{N} E_{p^t}[X^t_{i,j}]\right] \\
&= (1-\gamma)E[\sum_{t=1}^{T} p^t_i b^t_r] - \frac{N \ln N}{\gamma} - \frac{\gamma}{N} E\left[\sum_{t=1}^{T}\sum_{j=1}^{N} p^t_i b^t_j\right] \\
&\geq (1-\gamma)B_{i,r} - \frac{N \ln N}{\gamma} - \frac{\gamma}{N}\sum_{j=1}^{N} B_{i,j} \\
&\geq B_{i,r} - O(\sqrt{B_{max} N \ln N}) = B_{i,r} - R^{MAB}(B_{max}, N, T)\ ,
\end{aligned}$$

where $B_{i,r} = E[\sum_{t=1}^{T} p^t_i b^t_r]$, $B_{max} \geq \max_{i,j} B_{i,j}$ and $\gamma = \min\{\sqrt{(N \ln N)/B_{max}}, 1\}$.

For swap regret, we compare the expected benefit of *SR_MAB* to that of $\sum_{i=1}^{N} \max_j B_{i,j}$. Therefore,

$$B_{SR\_MAB} = \sum_{i=1}^{N} E[G_{A_i}] \geq \max_F \sum_{i=1}^{N} B_{i,F(i)} - N \cdot R^{MAB}(B_{max}, T, N)$$

which completes the proof of the theorem. ■

6. External Regret with Time-Selection Functions

We now present a simple online algorithm that achieves a good external regret bound in the presence of time selection functions, generalizing the *sleeping experts* setting. Specifically, our goal is for each action a, and each time-selection function I, that our total loss during the time-steps selected by I should be not much more than the loss of a during those time steps. More generally, this should be true for the losses *weighted* by I when $I(t) \in [0,1]$. The idea of the algorithm is as follows. Let $R_{a,I}$ be the regret of our algorithm with respect to action a and time selection function I. That is, $R_{a,I} = \sum_t I(t)(\ell^t_H - \ell^t_a)$. Let $\tilde{R}_{a,I}$ be a less-strict notion of regret in which we multiply our loss by some $\beta \in (0,1)$, that is, $\tilde{R}_{a,I} = \sum_t I(t)(\beta\ell^t_H - \ell^t_a)$. What we will do is give to each action a and time selection function I a weight $w_{a,I}$ that is exponential in $\tilde{R}_{a,I}$. We will prove that the sum of our weights never increases, and thereby be able to easily conclude that none of the $\tilde{R}_{a,I}$ can be too large.

Specifically, for each of the N actions and the M time selection functions we maintain a weight $w^t_{a,I}$. We update these weights using the rule $w^{t+1}_{a,I} = w^t_{a,I}\beta^{I(t)(\ell^t_a - \beta\ell^t_H)}$, where ℓ^t_H is the loss of our online algorithm H at time t. (Initially, $w^0_{a,I} = 1$.) Equivalently, $w^t_{a,I} = \beta^{-\tilde{R}^t_{a,I}}$, where $\tilde{R}^t_{a,I}$ is the "less-strict" regret mentioned above up to time t.

At time t we define $w^t_a = \sum_I I(t) w^t_{a,I}$, $W^t = \sum_a w^t_a$ and $p^t_a = w^t_a / W^t$. Our distribution over actions at time t is p^t. The following claim shows that the weights remain bounded.

Claim 12 *At any time t we have $0 \le \sum_{a,I} w_{a,I}^t \le NM$.*

Proof Initially, at time $t = 0$, the claim clearly holds. Observe that at time t we have the following identity,

$$W^t \ell_H^t = W^t \sum_a p_a^t \ell_a^t = \sum_a w_a^t \ell_a^t = \sum_a \sum_I I(t) w_{a,I}^t \ell_a^t. \tag{2}$$

For the inductive step we show that the sum of the weights can only decrease. Note that for any $\beta \in [0,1]$ and $x \in [0,1]$ we have $\beta^x \le 1-(1-\beta)x$ and $\beta^{-x} \le 1+(1-\beta)x/\beta$. Therefore,

$$\begin{aligned}
\sum_a \sum_I w_{a,I}^{t+1} &= \sum_a \sum_I w_{a,I}^t \beta^{I(t)(\ell_a^t - \beta \ell_H^t)} \\
&= \sum_a \sum_I w_{a,I}^t \beta^{I(t)\ell_a^t} \beta^{-\beta I(t)\ell_H^t)} \\
&\le \sum_a \sum_I w_{a,I}^t (1-(1-\beta)I(t)\ell_a^t)(1+(1-\beta)I(t)\ell_H^t) \\
&\le \left(\sum_a \sum_I w_{a,I}^t\right) - (1-\beta)\left(\sum_{a,I} I(t) w_{a,I}^t \ell_a^t\right) + (1-\beta)\left(\sum_{a,I} I(t) w_{a,I}^t \ell_H^t\right) \\
&= \left(\sum_a \sum_I w_{a,I}^t\right) - (1-\beta) W^t \ell_H^t + (1-\beta) W^t \ell_H^t \quad \text{(using Equation 2)} \\
&= \left(\sum_a \sum_I w_{a,I}^t\right) \quad ,
\end{aligned}$$

which completes the proof of the claim. ■

We use the above claim to bound the weight of any action a and time-selection function I.

Corollary 13 *For every action a and time selection I we have*

$$w_{a,I}^t = \beta^{L_{a,I} - \beta L_{H,I}} \le MN,$$

where $L_{H,I} = \sum_t I(t)\ell_H^t$ is the loss of the online algorithm with respect to time-selection function I.

A simple algebraic manipulation of the above implies the following theorem

Theorem 14 *For every action a and every time selection function $I \in I$ we have*

$$L_{H,I} \le \frac{L_{a,I} + \frac{\log NM}{\log \frac{1}{\beta}}}{\beta} .$$

We can optimize for β in advance, or do it dynamically using Auer et al. (2002b), establishing:

Corollary 15 *For every action a and every time selection function $I \in I$ we have*

$$L_{H,I} \le L_{a,I} + O(\sqrt{L_{min} \log NM} + \log MN),$$

where $L_{min} = \max_I \min_a \{L_{a,I}\}$.

One can get a more refined regret bound of $O(\sqrt{L_{min,I}\log NM}+\log MN)$ with respect to each time selection function $I \in \mathcal{I}$, where $L_{min,I} = \min_a\{L_{a,I}\}$. This is achieved by keeping a parameter β_I for each time selection function $I \in \mathcal{I}$. As before we then set $w^t_{a,I} = \beta_I^{-\tilde{R}^t_{a,I}}$, where $\tilde{R}^t_{a,I} = \sum_{t'\le t} I(t')(\beta_I \ell^{t'}_H - \ell^{t'}_a)$. We then let $w^t_a = \sum_I (1-\beta_I) I(t) w^t_{a,I}$, $W^t = \sum_a w^t_a$ and $p^t_a = w^t_a/W^t$. The proof of Claim 12 holds in a similar way, and from that one can derive, analogously, the more refined regret bound, as stated in the following theorem.

Theorem 16 *For every action a and every time selection function $I \in \mathcal{I}$ we have*

$$L_{H,I} \le L_{a,I} + O(\sqrt{L_{min,I}\log NM}+\log MN),$$

where $L_{min,I} = \min_a\{L_{a,I}\}$.

7. Arbitrary Time Selection and Modification Rules

In this section we combine the techniques from Sections 3 and 6 to derive a regret bound for the general case where we assume that there is a finite set $\mathcal{I}$ of M time selection functions, and a finite set $\mathcal{F}$ of K modification rules. Our goal is to design an algorithm such that for any time selection function $I \in \mathcal{I}$ and any $F \in \mathcal{F}$, we have that $L_{H,I}$ is not much larger than $L_{H,I,F}$. Essentially, our results can be viewed as deriving explicit rates for the wide range regret of Lehrer (2003).

We maintain at time t a weight $w^t_{j,I,F}$ per action j, time selection I and modification rule F. Initially $w^0_{j,I,F} = 1$. We set

$$w^{t+1}_{j,I,F} = w^t_{j,I,F}\beta^{p^t_j I(t)(\ell^t_{F(j)} - \beta\ell^t_{H,j})},$$

and let $W^t_{j,F} = \sum_I I(t) w^t_{j,I,F}$, $W^t_j = \sum_F W^t_{j,F}$, and $\ell^t_{H,j} = \sum_F W^t_{j,F}\ell^t_{F(j)}/W^t_j$.

We use the weights to define a distribution p^t over actions as follows. We select a distribution p^t such that

$$p^t_i = \sum_{j=1}^N p^t_j \sum_{F:F(j)=i} \frac{W^t_{j,F}}{W^t_j}. \tag{3}$$

That is, p is the stationary distribution of the associated Markov chain. Notice that the definition of p implies that the loss of H at time t can either be viewed as $\sum_i p^t_i \ell^t_i$ or as $\sum_j p_j \sum_F (W^t_{j,F}/W^t_j)\ell^t_{F(j)} = \sum_j p^t_j \ell^t_{H,j}$. The following claim bounds the magnitude of the weights.

Claim 17 *For every action j, at any time t we have* $0 \le \sum_{I,F} w^t_{j,I,F} \le MK$.

Proof This clearly holds initially at $t=0$. For any $t \ge 0$ we show that $\sum_{I,F} w^{t+1}_{j,I,F} \le \sum_{I,F} w^t_{j,I,F}$. Recall that for $\beta \in [0,1]$ and $x \in [0,1]$ we have $\beta^x \le 1-(1-\beta)x$ and $\beta^{-x} \le 1+(1-\beta)x/\beta$.

$$\begin{aligned}
\sum_{I,F} w^{t+1}_{j,I,F} &= \sum_{I,F} w^t_{j,I,F}\beta^{p^t_j I(t)(\ell^t_{F^t(j)} - \beta\ell^t_{H,j})} \\
&\le \sum_{I,F} w^t_{j,I,F}(1-(1-\beta)p^t_j I(t)\ell^t_{F^t(j)})(1+(1-\beta)p^t_j I(t)\ell^t_{H,j}) \\
&\le \left(\sum_{I,F} w^t_{j,I,F}\right) - (1-\beta)p^t_j \sum_F \ell^t_{F^t(j)} \sum_I I(t) w^t_{j,I,F} + (1-\beta)p^t_j \ell^t_{H,j}\sum_{I,F} I(t) w^t_{j,I,F}
\end{aligned}$$

$$\begin{aligned}
&= \left(\sum_{I,F} w^t_{j,I,F}\right) - (1-\beta)p^t_j \sum_F \ell^t_{F^t(j)} W^t_{j,F} + (1-\beta)p^t_j \ell^t_{H,j} W^t_j \\
&= \left(\sum_{I,F} w^t_{j,I,F}\right) - (1-\beta)p^t_j W^t_j \ell^t_{H,j} + (1-\beta)p^t_j W^t_j \ell^t_{H,j} \\
&= \sum_{I,F} w^t_{j,I,F} \ ,
\end{aligned}$$

where in the second to last equality we used the identity $\sum_F \ell^t_{F^t(j)} W^t_{j,F} = \ell^t_{H,j} W^t_j$. ■

The following theorem derives the general regret bound.

Theorem 18 *For every time selection $I \in \mathcal{I}$ and modification rule $F \in \mathcal{F}$, we have that*

$$L_{H,I} \le L_{H,I,F} + O(\sqrt{TN \log MK}) \ .$$

Proof Consider a time selection function $I \in \mathcal{I}$ and a modification function $F \in \mathcal{F}$. By Claim 17 we have that,

$$w^T_{j,I,F} = \beta^{(\sum_t p^t_j I(t) \ell^t_{F^t(j)}) - \beta(\sum_t p^t_j I(t) \ell^t_{H,j})} \le MK \ ,$$

which is equivalent to

$$\left(\sum_t I(t) p^t_j \ell^t_{H,j}\right) \le \frac{1}{\beta}\left(\sum_t I(t) p^t_j \ell^t_{F^t(j)}\right) + \frac{\log MK}{\beta \log \frac{1}{\beta}} \ .$$

Notice that $\sum_{j,t} I(t) p^t_j \ell^t_{H,j} = \sum_{i,t} I(t) p^t_i \ell^t_i$, by the definition of the p_i's in Equation (3). Summing over all actions j this sum is $L_{H,I}$. Therefore,

$$\begin{aligned}
L_{H,I} = \sum_{j=1}^{N}\left(\sum_t I(t) p^t_j \ell^t_{H,j}\right) &\le \sum_{j=1}^{N} \frac{1}{\beta}\left(\sum_t I(t) p^t_j \ell^t_{F^t(j)}\right) + \frac{N \log MK}{\beta \log \frac{1}{\beta}} \\
&= \frac{1}{\beta} L_{H,I,F} + \frac{N \log MK}{\beta \log \frac{1}{\beta}},
\end{aligned}$$

where $L_{H,I}$ is the cost of the online algorithm at time selection I and $L_{H,I,F}$ is the cost of the modified output sequence at time selection I. Optimizing for β we derive the theorem. ■

8. Boolean Prediction with Time Selection

In this section we consider the case that there are two actions $\{0,1\}$, and the loss function is such that at every time step t one action has loss 1 and the other has loss 0. Namely, we assume that the adversary returns at time t an action $o^t \in \{0,1\}$, and the loss of action a^t is 1 if $a^t \neq o^t$ and 0 if $a^t = o^t$. Our objective here is to achieve good bounds with a deterministic algorithm.

For each time selection function $I \in \mathcal{I}$, action $a \in \{0,1\}$, and time t, our online Boolean prediction algorithm maintains a weight $w^t_{a,I}$. Initially we set $w^0_{a,I} = 1$ for every action $a \in \{0,1\}$ and time selection function $I \in \mathcal{I}$. At time t, for each action $a \in \{0,1\}$, we compute $w^t_a = \sum_I I(t) w^t_{a,I}$, and predict $a^t = 1$ if $w_1^t \geq w_0^t$, and otherwise predict $a^t = 0$. The *weighted errors* of our online Boolean prediction algorithm, during the time selection function $I \in \mathcal{I}$, is $\sum_{t:o_t \neq a^t} I(t)$.

Following our prediction we observe the adversary action o^t. If no error occurred (i.e., $a_t = o_t$) then all the weights at time $t+1$ equal the weights at time t. If an error occurred (i.e., $a_t \neq o_t$) then we update the weights as follows. For every time selection function $I \in \mathcal{I}$ we set the weight of action b to $w_{b,I}^{t+1} = w_{b,I}^t 2^{cI(t)}$, where $c = -1$ if $b \neq o_t$ and $c = 1/2$ if $b = o_t$. This can be viewed as a version of the Balanced-Winnow algorithm of Littlestone (1988). We establish the following claim,

Claim 19 *At any time t we have* $0 \leq \sum_{a,I} w_{a,I}^t \leq 2M$.

Proof Clearly this holds at time $t = 0$. When an error is performed, we have that $w_{error}^t \geq w_{correct}^t$, where $correct = o^t$ and $error = 1 - o^t$. The additive change in the weights is at most $(\sqrt{2}-1)w_{correct}^t - w_{error}^t/2 < 0$, which completes the proof. ■

For a time selection function $I \in \mathcal{I}$, let $v_{a,I} = \sum_{t:o_t=a} I(t)$. The *preferred action* for a time selection function I is 1 if $v_{1,I} \geq v_{0,I}$ and 0 otherwise. Let $OPT(I)$ be the weighted errors of the preferred action during time selection function I. W.l.o.g., assume that the preferred action for I is 1, which implies that $OPT(I) = v_{0,I}$. By Claim 19 we have that $w_{1,I}^t \leq 2M$. The total decrease in $w_{1,I}^t$ is bounded by a factor of $2^{-v_{0,I}}$. Since $w_{1,I}^T \leq 2M$ this implies that $2^{x/2-v_{0,I}} \leq 2M$, where x is the total increase, which implies that

$$x \leq 2v_{0,I} + 2 + 2\log_2 M\ .$$

The weighted errors of our online Boolean prediction algorithm, during time selection function $I \in \mathcal{I}$, that is, $\sum_{t:a^t \neq o^t} I(t)$, is at most $x + v_{0,I}$, while the preferred action makes only $v_{0,I}$ weighted errors. This implies that the weighted errors of our online Boolean prediction algorithm during time selection function I is bounded by $x + v_{0,I} = 3v_{0,I} + 2 + 2\log_2 M$, which establishes the following theorem.

Theorem 20 *For every $I \in \mathcal{I}$, our online algorithm makes at most $3OPT(I) + 2 + 2\log_2 M$ weighted errors.*

9. Conclusion and Open Problems

In this paper we give general reductions by which algorithms achieving good external regret can be converted to algorithms with good internal or swap regret, and in addition we develop algorithms for a generalization of the sleeping experts scenario including both real-valued time-selection functions and a finite set of modification rules.

A key problem left open by this work is whether it is possible to achieve swap-regret that has a sublinear or even logarithmic dependence on N. Specifically, for *external* regret, existing algorithms achieve regret εT in time $T = O(\frac{1}{\varepsilon^2}\log N)$, but our algorithms for swap-regret achieve regret εT only by time $T = O(\frac{1}{\varepsilon^2} N \log N)$. We have shown that sublinear dependence is not possible against an adaptive adversary with swap-regret defined with respect to the actions actually chosen from the algorithm's distribution, but we do not know whether there is a comparable lower bound in the distributional setting (where swap-regret is defined with respect to the distributions p^t themselves). In particular, an algorithm with lower dependence on N would imply a more efficient (in terms of number of rounds) procedure for achieving an approximate correlated equilibrium. Ideally, one would like to achieve approximate correlated equilibrium in a number of rounds that is only logarithmic in N, much as can be done for approximate minimax optimality in 2-player zero-sum games.

Acknowledgments

We would like to thank Gilles Stoltz for a number of helpful comments and suggestions.

References

P. Auer, N. Cesa-Bianchi, Y. Freund, and R.E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002a.

P. Auer, N. Cesa-Bianchi, and C. Gentile. Adaptive and self-confident on-line learning algorithms. *Journal of Computing and System Sciences*, 64(1):48–75, 2002b.

R. J. Aumann. Subjectivity and correlation in randomized strategies. *Journal of Mathematical Economics*, 1:67–96, 1974.

D. Blackwell. An analog of the mimimax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6:1–8, 1956.

A. Blum. Empirical support for Winnow and Weighted-Majority based algorithms: Results on a calendar scheduling domain. *Machine Learning*, 26:5–23, 1997.

N. Cesa-Bianchi, Y. Freund, D.P. Helmbold, D. Haussler, R.E. Schapire, and M.K. Warmuth. How to use expert advice. *Journal of the Association for Computing Machinery (JACM)*, 44(3):427–485, 1997.

N. Cesa-Bianchi and G. Lugosi. Potential-based algorithms in on-line prediction and game theory. *Machine Learning*, 51(3):239–261, 2003.

N. Cesa-Bianchi, G. Lugosi, and G. Stoltz. Regret minimization under partial monitoring. *Mathematics of Operations Research*, 31:562–580, 2006.

N. Cesa-Bianchi, Y. Mansour, and G. Stoltz. Improved second-order bounds for prediction with expert advice. In *Proceedings of the Eighteenth Annual Conference on Computational Learning Theory*, pages 217–232, 2005.

W. Cohen and Y. Singer. Learning to query the web. In *AAAI Workshop on Internet-Based Information Systems*, 1996.

W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. *ACM Transactions on Information Systems*, 17(2):141–173, 1999.

W. Feller. *An Introduction to Probability Theory and its Applications. - Vol. 1.* Wiley, 1968.

W. Feller. *An Introduction to Probability Theory and its Applications. - Vol. 2.* Wiley, 1971.

D. Foster and R. Vohra. A randomization rule for selecting forecasts. *Operations Research*, 41(4): 704–709, July–August 1993.

D. Foster and R. Vohra. Calibrated learning and correlated equilibrium. *Games and Economic Behavior*, 21:40–55, 1997.

D. Foster and R. Vohra. Asymptotic calibration. *Biometrika*, 85:379–390, 1998.

D. Foster and R. Vohra. Regret in the on-line decision problem. *Games and Economic Behavior*, 29:7–36, 1999.

Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

Y. Freund and R.E. Schapire. Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, 29:79–103, 1999.

Y. Freund, R.E. Schapire, Y. Singer, and M.K. Warmuth. Using and combining predictors that specialize. In *Proceedings of the 29th Annual Symposium on Theory of Computing*, pages 334–343, 1997.

J. Hannan. Approximation to Bayes risk in repeated plays. In M. Dresher, A. Tucker, and P. Wolfe, editors, *Contributions to the Theory of Games*, volume 3, pages 97–139. Princeton University Press, 1957.

S. Hart and A. Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68:1127–1150, 2000.

S. Hart and A. Mas-Colell. A reinforcement procedure leading to correlated equilibrium. In Wilhelm Neuefeind Gerard Debreu and Walter Trockel, editors, *Economic Essays*, pages 181–200. Springer, 2001.

E. Lehrer. A wide range no-regret theorem. *Games and Economic Behavior*, 42:101–115, 2003.

N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

N. Littlestone and M.K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108:212–261, 1994.

G. Stoltz. *Incomplete Information and Internal Regret in Prediction of Individual Sequences*. PhD thesis, Dept. of Mathematics, University Paris XI, ORSAY, 2005.

G. Stoltz and G. Lugosi. Internal regret in on-line portfolio selection. *Machine Learning*, 59(1-2): 125–159, 2005.

G. Stoltz and G. Lugosi. Learning correlated equilibria in games with compact sets of strategies. *Games and Economic Behavior*, 59:187–209, 2007.

Journal of Machine Learning Research 8 (2007) 1325-1368 Submitted 5/06; Revised 3/07; Published 5/07

Graph Laplacians and their Convergence on Random Neighborhood Graphs

Matthias Hein MH@TUEBINGEN.MPG.DE
Max Planck Institute for Biological Cybernetics
Dept. Schölkopf
Spemannstraße 38
Tübingen 72076, Germany

Jean-Yves Audibert AUDIBERT@CERTIS.FR
CERTIS
Ecole Nationale des Ponts et Chaussées
19, rue Alfred Nobel - Cité Descartes
F-77455 Marne-la-Vallée cedex 2, France

Ulrike von Luxburg ULRIKE.LUXBURG@TUEBINGEN.MPG.DE
Max Planck Institute for Biological Cybernetics
Dept. Schölkopf
Spemannstraße 38
Tübingen 72076, Germany

Editor: Sanjoy Dasgupta

Abstract

Given a sample from a probability measure with support on a submanifold in Euclidean space one can construct a neighborhood graph which can be seen as an approximation of the submanifold. The graph Laplacian of such a graph is used in several machine learning methods like semi-supervised learning, dimensionality reduction and clustering. In this paper we determine the pointwise limit of three different graph Laplacians used in the literature as the sample size increases and the neighborhood size approaches zero. We show that for a uniform measure on the submanifold all graph Laplacians have the same limit up to constants. However in the case of a non-uniform measure on the submanifold only the so called random walk graph Laplacian converges to the weighted Laplace-Beltrami operator.

Keywords: graphs, graph Laplacians, semi-supervised learning, spectral clustering, dimensionality reduction

1. Introduction

In recent years, methods based on graph Laplacians have become increasingly popular in machine learning. They have been used in semi-supervised learning (Belkin and Niyogi, 2004; Zhou et al., 2004; Zhu and Ghahramani, 2002), spectral clustering (Spielman and Teng, 1996; von Luxburg, 2006) and dimensionality reduction (Belkin and Niyogi, 2003; Coifman and Lafon, 2006). Their popularity is mainly due to the following properties of the Laplacian which will be discussed in more detail in a later section:

- the Laplacian is the generator of the diffusion process (label propagation in semi-supervised learning),
- the eigenvectors of the Laplacian have special geometric properties (motivation for spectral clustering),
- the Laplacian induces an adaptive regularization functional, which adapts to the density and the geometric structure of the data (semi-supervised learning, classification).

If the data lies in $\mathbb{R}^d$ the neighborhood graph built from the random sample can be seen as an approximation of the continuous structure. In particular, if the data has support on a low-dimensional submanifold the neighborhood graph is a discrete approximation of the submanifold. In machine learning we are interested in the intrinsic properties and objects of this submanifold. The approximation of the Laplace-Beltrami operator via the graph Laplacian is a very important one since it has numerous applications as we will discuss later.

Approximations of the Laplace-Beltrami operator or related objects have been studied for certain special deterministic graphs. The easiest case is a grid in $\mathbb{R}^d$. In numerics it is standard to approximate the Laplacian with finite-differences schemes on the grid. These can be seen as a special instances of a graph Laplacian. There convergence for decreasing grid-size follows easily by an argument using Taylor expansions. Another more involved example is the work of Varopoulos (1984), where for a graph generated by an ε-packing of a manifold, the equivalence of certain properties of random walks on the graph and Brownian motion on the manifold have been established. The connection between random walks and the graph Laplacian becomes obvious by noting that the graph Laplacian as well as the Laplace-Beltrami operator are the generators of the diffusion process on the graph and the manifold, respectively. In Xu (2004) the convergence of a discrete approximation of the Laplace Beltrami operator for a triangulation of a 2D-surface in $\mathbb{R}^3$ was shown. However, it is unclear whether the approximation described there can be written as a graph Laplacian and whether this result can be generalized to higher dimensions.

In the case where the graph is generated randomly, only first results have been proved so far. The first work on the large sample limit of graph Laplacians has been done by Bousquet et al. (2004). There the authors studied the convergence of the regularization functional induced by the graph Laplacian using the law of large numbers for U-statistics. In a second step taking the limit of the neighborhoodsize $h \to 0$, they got $\frac{1}{p^2}\nabla(p^2\nabla)$ as the effective limit operator in $\mathbb{R}^d$. Their result has recently been generalized to the submanifold case and uniform convergence over the space of Hölder-functions by Hein (2005, 2006). In von Luxburg et al. (2007), the neighborhoodsize h was kept fixed while the large sample limit of the graph Laplacian was considered. In this setting, the authors showed strong convergence results of graph Laplacians to certain integral operators, which imply the convergence of the eigenvalues and eigenfunctions. Thereby showing the consistency of spectral clustering for a fixed neighborhood size.

In contrast to the previous work in this paper we will consider the large sample limit and the limit as the neighborhood size approaches zero simultaneously for a certain class of neighbhorhood graphs. The main emphasis lies on the case where the data generating measure has support on a submanifold of $\mathbb{R}^d$. The bias term, that is the difference between the continuous counterpart of the graph Laplacian and the Laplacian itself has been studied first for compact submanifolds without boundary by Smolyanov, von Weizsäcker, and Wittich (2000) and Belkin (2003) for the Gaussian kernel and a uniform data generating measure and was then generalized by Lafon (2004) to general isotropic weights and general probability measures. Additionally Lafon showed that the use of data-dependent weights for the graph allows to control the influence of the density. They all show that the bias term converges pointwise if the neighborhood size goes to zero. The convergence of the graph Laplacian towards these continuous averaging operators was left open. This part was first studied by Hein et al. (2005) and Belkin and Niyogi (2005). In Belkin and Niyogi (2005) the convergence was shown for the so called unnormalized graph Laplacian in the case of a uniform probability measure on a compact manifold without boundary and using the Gaussian kernel for the weights, whereas in Hein et al. (2005) the pointwise convergence was shown for the random walk graph Laplacian in the case of general probability measures on non-compact manifolds with boundary using general isotropic data-dependent weights. More recently Giné and Koltchinskii (2006) have extended the pointwise convergence for the unnormalized graph Laplacian shown by Belkin and Niyogi (2005) to uniform convergence on compact submanifolds without boundary giving explicit rates. In Singer (2006), see also Giné and Koltchinskii (2006), the rate of convergence given by Hein et al. (2005) has been improved in the setting of the uniform measure. In this paper we will study the three most often used graph Laplacians in the machine learning literature and show their pointwise convergence in the general setting of Lafon (2004) and Hein et al. (2005), that is we will in particular consider the case where by using data-dependent weights for the graph we can control the influence of the density on the limit operator.

In Section 2 we introduce the basic framework necessary to define graph Laplacians for general directed weighted graphs and then simplify the general case to undirected graphs. In particular, we define the three graph Laplacians used in machine learning so far, which we call the normalized, the unnormalized and the random walk Laplacian. In Section 3 we introduce the neighborhood graphs studied in this paper, followed by an introduction to the so called weighted Laplace-Beltrami operator, which will turn out to be the limit operator in general. We also study properties of this limit operator and provide insights why and how this operator can be used for semi-supervised learning, clustering and regression. Then finally we present the main convergence result for all three graph Laplacians and give the conditions on the neighborhood size as a function of the sample size necessary for convergence. In Section 4 we illustrate the main result by studying the difference between the three graph Laplacians and the effects of different data-dependent weights on the limit operator. In Section 5 we prove the main result. We introduce a framework for studying non-compact manifolds with boundary and provide the necessary assumptions on the submanifold M, the data generating measure P and the kernel k used for defining the weights of the edges. We would like to note that the theorems given in Section 5 contain slightly stronger results than the ones presented in Section 3. The reader who is not familiar with differential geometry will find a brief introduction to the basic material used in this paper in Appendix A.

2. Abstract Definition of the Graph Structure

In this section we define the structure on a graph which is required in order to define the graph Laplacian. To this end one has to introduce Hilbert spaces H_V and H_E of functions on the vertices V and edges E, define a difference operator d, and then set the graph Laplacian as $\Delta = d^*d$. We first do this in full generality for directed graphs and then specialize it to undirected graphs. This approach is well-known for undirected graphs in discrete potential theory and spectral graph theory, see for example Dodziuk (1984), Chung (1997), Woess (2000) and McDonald and Meyers (2002), and was generalized to directed graphs by Zhou et al. (2005) for a special choice of H_V, H_E and d. To our knowledge the very general setting introduced here has not been discussed elsewhere.

In many articles graph Laplacians are used without explicitly mentioning d, H_V and H_E. This can be misleading since, as we will show, there always exists a whole family of choices for d, H_V and H_E which all yield the same graph Laplacian.

2.1 Hilbert Spaces of Functions on the Vertices V and the Edges E

Let (V,W) be a graph where V denotes the set of vertices with $|V| = n$ and W a positive $n \times n$ similarity matrix, that is $w_{ij} \geq 0$, $i, j = 1,\dots,n$. W need not to be symmetric, that means we consider the case of a directed graph. The special case of an undirected graph will be discussed in a following section. Let $E \subset V \times V$ be the set of edges $e_{ij} = (i, j)$ with $w_{ij} > 0$. e_{ij} is said to be a directed edge from the vertex i to the vertex j with weight w_{ij}. Moreover, we define the outgoing and ingoing sum of weights of a vertex i as

$$d_i^{out} = \frac{1}{n}\sum_{j=1}^{n} w_{ij}, \quad d_i^{in} = \frac{1}{n}\sum_{j=1}^{n} w_{ji}.$$

We assume that $d_i^{out} + d_i^{in} > 0$, $i = 1,\dots,n$, meaning that each vertex has at least one in- or outgoing edge. Let $\mathbb{R}_+ = \{x \in \mathbb{R} \,|\, x \geq 0\}$ and $\mathbb{R}_+^* = \{x \in \mathbb{R} \,|\, x > 0\}$. The inner product on the function space $\mathbb{R}^V$ is defined as

$$\langle f, g\rangle_V = \frac{1}{n}\sum_{i=1}^{n} f_i g_i \chi_i,$$

where $\chi_i = (\chi_{out}(d_i^{out}) + \chi_{in}(d_i^{in}))$ with $\chi_{out} : \mathbb{R}_+ \to \mathbb{R}_+$ and $\chi_{in} : \mathbb{R}_+ \to \mathbb{R}_+$, $\chi_{out}(0) = \chi_{in}(0) = 0$ and further χ_{out} and χ_{in} strictly positive on $\mathbb{R}_+^*$.

We also define an inner product on the space of functions $\mathbb{R}^E$ on the edges:

$$\langle F, G\rangle_E = \frac{1}{2n^2}\sum_{i,j=1}^{n} F_{ij} G_{ij} \phi(w_{ij}),$$

where $\phi : \mathbb{R}_+ \to \mathbb{R}_+$, $\phi(0) = 0$ and ϕ strictly positive on $\mathbb{R}_+^*$. Note that with these assumptions on ϕ the sum is taken only over the set of edges E. One can check that both inner products are well-defined. We denote by $\mathcal{H}(V,\chi) = (\mathbb{R}_V, \langle\cdot,\cdot\rangle_V)$ and $\mathcal{H}(E,\phi) = (\mathbb{R}^E, \langle\cdot,\cdot\rangle_E)$ the corresponding Hilbert spaces. As a last remark let us clarify the roles of $\mathbb{R}^V$ and $\mathbb{R}^E$. The first one is the space of functions on the vertices and therefore can be regarded as a normal function space. However, elements of $\mathbb{R}^E$ can be interpreted as a flow on the edges so that the function value on an edge e_{ij} corresponds to the "mass" flowing from one vertex i to the vertex j (per unit time).

2.2 The Difference Operator d and its Adjoint d^*

Definition 1 *The* ***difference operator*** $d : \mathcal{H}(V,\chi) \to \mathcal{H}(E,\phi)$ *is defined as follows:*

$$\forall\, e_{ij} \in E, \qquad (df)(e_{ij}) = \gamma(w_{ij})\,(f(j) - f(i)),$$

where $\gamma : \mathbb{R}_+^* \to \mathbb{R}_+^*$.

Remark 2 *Note that d is zero on the constant functions as one would expect it from a derivative. In Zhou et al. (2004) another difference operator d is used:*

$$(df)(e_{ij}) = \gamma(w_{ij})\Big(\frac{f(j)}{\sqrt{d(j)}} - \frac{f(i)}{\sqrt{d(i)}}\Big). \tag{1}$$

Note that in Zhou et al. (2004) they have $\gamma(w_{ij}) \equiv 1$. This difference operator is in general not zero on the constant functions. This in turn leads to the effect that the associated Laplacian is also not zero on the constant functions. For general graphs without any geometric interpretation this is just a matter of choice. However, the choice of d matters if one wants a consistent continuum limit of the graph. One cannot expect convergence of the graph Laplacian associated to the difference operator d of Equation (1) towards a Laplacian, since as each of the graph Laplacians in the sequence is not zero on the constant functions, also the limit operator will share this property unless $\lim_{n\to\infty} d(X_i) = c, \forall i = 1,\dots,n$, where c is a constant. We derive also the limit operator of the graph Laplacian induced by the difference operator of Equation (1) introduced by Zhou et al. in the machine learning literature and usually denoted as the normalized graph Laplacian in spectral graph theory (Chung, 1997).

Obviously, in the finite case d is always a bounded operator. The adjoint operator $d^* : \mathcal{H}(E,\phi) \to \mathcal{H}(V,\chi)$ is defined by

$$\langle df, u\rangle_E = \langle f, d^*u\rangle_V, \quad \forall f \in H(V,\chi), \quad u \in \mathcal{H}(E,\phi).$$

Lemma 3 *The adjoint $d^* : \mathcal{H}(E,\phi) \to \mathcal{H}(V,\chi)$ of the difference operator d is explicitly given by:*

$$(d^*u)(l) = \frac{1}{2\chi_l}\Big(\frac{1}{n}\sum_{i=1}^n \gamma(w_{il})\, u_{il}\, \phi(w_{il}) - \frac{1}{n}\sum_{i=1}^n \gamma(w_{li})\, u_{li}\, \phi(w_{li})\Big). \tag{2}$$

Proof Using the indicator function $f(j) = \mathbb{1}_{j=l}$ it is straightforward to derive:

$$\frac{1}{n}\chi_l\,(d^*u)(l) = \langle d\mathbb{1}_{\cdot=l}, u\rangle_E = \frac{1}{2n^2}\sum_{i=1}^n \Big(\gamma(w_{il})u_{il}\phi(w_{il}) - \gamma(w_{li})u_{li}\phi(w_{li})\Big),$$

where we have used $\langle d\mathbb{1}_{\cdot=l}, u\rangle_E = \frac{1}{2n^2}\sum_{i,j=1}^n (d\mathbb{1}_{\cdot=l})_{ij} u_{ij}\phi(w_{ij})$. ■

The first term of the rhs of (2) can be interpreted as the outgoing flow, whereas the second term can be seen as the ingoing flow. The corresponding continuous counterpart of d is the gradient of a function and for d^* it is the divergence of a vector-field, measuring the infinitesimal difference between in- and outgoing flow.

2.3 The General Graph Laplacian

Definition 4 (graph Laplacian for a directed graph) *Given Hilbert spaces $\mathcal{H}(V,\chi)$ and $\mathcal{H}(E,\phi)$ and the difference operator $d : \mathcal{H}(V,\chi) \to \mathcal{H}(E,\phi)$ the graph Laplacian $\Delta : \mathcal{H}(V,\chi) \to \mathcal{H}(V,\chi)$ is defined as*

$$\Delta = d^*d.$$

Lemma 5 *Explicitly,* $\Delta : \mathcal{H}(V,\chi) \to \mathcal{H}(V,\chi)$ *is given as:*

$$(\Delta f)(l) = \frac{1}{2\chi_l}\left[\frac{1}{n}\sum_{i=1}^{n}\left(\gamma(w_{il})^2\phi(w_{il}) + \gamma(w_{li})^2\phi(w_{li})\right)\left(f(l)-f(i)\right)\right].$$

Proof The explicit expression Δ can be easily derived by plugging the expression of d^* and d together:

$$\begin{aligned}(d^*df)(l) =& \frac{1}{2\chi_l}\left(\frac{1}{n}\sum_{i=1}^{n}\gamma(w_{il})^2[f(l)-f(i)]\phi(w_{il}) - \frac{1}{n}\sum_{i=1}^{n}\gamma(w_{li})^2[f(i)-f(l)]\phi(w_{li})\right)\\ =& \frac{1}{2\chi_l}\left[f(l)\frac{1}{n}\sum_{i=1}^{n}\widehat{w}_{ij} - \frac{1}{n}\sum_{i=1}^{n}f(i)\widehat{w}_{ij}\right],\end{aligned}$$

where we have introduced $\widehat{w}_{ij} = \left(\gamma(w_{il})^2\phi(w_{il}) + \gamma(w_{li})^2\phi(w_{li})\right)$. ■

Proposition 6 Δ *is self-adjoint and positive semi-definite.*

Proof By definition, $\langle f, \Delta g\rangle_V = \langle df, dg\rangle_E = \langle \Delta f, g\rangle_V$, and $\langle f, \Delta f\rangle_V = \langle df, df\rangle_E \geq 0$. ■

2.4 The Special Case of an Undirected Graph

In the case of an undirected graph we have $w_{ij} = w_{ji}$, that is whenever there is an edge from i to j there is an edge with the same value from j to i. This implies that there is no difference between in- and outgoing edges. Therefore, $d_i^{out} \equiv d_i^{in}$, so that we will denote the degree function by d with $d_i = \frac{1}{n}\sum_{j=1}^{n} w_{ij}$. The same for the weights in H_V, $\chi_{out} \equiv \chi_{in}$, so that we have only one function χ. If one likes to interpret functions on E as flows, it is reasonable to restrict the space $\mathcal{H}_E$ to antisymmetric functions since symmetric functions are associated to flows which transport the same mass from vertex i to vertex j and back. Therefore, as a net effect, no mass is transported at all so that from a physical point of view these functions cannot be observed at all. Since anyway we consider only functions on the edges of the form df (where f is in $\mathcal{H}_V$) which are by construction antisymmetric, we will not do this restriction explicitly.

The adjoint d^* simplifies in the undirected case to

$$(d^*u)(l) = \frac{1}{2\chi(d_l)}\frac{1}{n}\sum_{i=1}^{n}\gamma(w_{il})\phi(w_{il})(u_{il} - u_{li}),$$

and the general graph Laplacian on an undirected graph has the following form:

Definition 7 (graph Laplacian for an undirected graph) *Given Hilbert spaces* $\mathcal{H}(V,\chi)$ *and* $\mathcal{H}(E,\phi)$ *and the difference operator* $d : \mathcal{H}(V,\chi) \to \mathcal{H}(E,\phi)$ *the graph Laplacian* $\Delta : \mathcal{H}(V,\chi) \to \mathcal{H}(V,\chi)$ *is defined as*

$$\Delta = d^*d.$$

Explicitly, for any vertex l*, we have*

$$(\Delta f)(l) = (d^*df)(l) = \frac{1}{\chi(d_l)}\left[f(l)\frac{1}{n}\sum_{i=1}^{n}\gamma^2(w_{il})\phi(w_{il}) - \frac{1}{n}\sum_{i=1}^{n}f(i)\gamma^2(w_{il})\phi(w_{il})\right].$$

In the literature one finds the following special cases of the general graph Laplacian. Unfortunately there exist no unique names for the three graph Laplacians we introduce here, most of the time all of them are just called graph Laplacians. Only the term ’unnormalized’ or ’combinatorial’ graph Laplacian seems to be established now. However, the other two could both be called normalized graph Laplacian. Since the first one is closely related to a random walk on the graph we call it random walk graph Laplacian and the other one normalized graph Laplacian.

The ’random walk’ graph Laplacian is defined as:

$$(\Delta^{(\mathrm{rw})} f)(i) = f(i) - \frac{1}{d_i}\frac{1}{n}\sum_{j=1}^{n} w_{ij} f(j), \quad \Delta^{(\mathrm{rw})} f = (\mathbb{1} - D^{-1}W) f,$$

where the matrix D is defined as $D_{ij} = d_i \delta_{ij}$. Note that $P = D^{-1}W$ is a stochastic matrix and therefore can be used to define a Markov random walk on V, see for example Woess (2000). The ’unnormalized’ (or ’combinatorial’) graph Laplacian is defined as

$$(\Delta^{(\mathrm{u})} f)(i) = d_i f(i) - \frac{1}{n}\sum_{j=1}^{n} w_{ij} f(j), \quad \Delta^{(\mathrm{u})} f = (D - W) f.$$

We have the following conditions on χ, γ and ϕ in order to get these Laplacians:

$$\forall e_{ij} \in E: \quad \mathrm{rw}: \quad \frac{\gamma^2(w_{ij})\phi(w_{ij})}{\chi(d_i)} = \frac{w_{ij}}{d_i}, \quad \mathrm{unnorm}: \quad \frac{\gamma^2(w_{ij})\phi(w_{ij})}{\chi(d_i)} = w_{ij}.$$

We observe that by choosing $\Delta^{(\mathrm{rw})}$ or $\Delta^{(\mathrm{u})}$ the functions ϕ and γ are not fixed. Therefore it can cause confusion if one speaks of the ’random walk’ or ’unnormalized’ graph Laplacian without explicitly defining the corresponding Hilbert spaces and the difference operator. We also consider the normalized graph Laplacian $\Delta^{(\mathrm{n})}$ introduced by Chung (1997); Zhou et al. (2004) using the difference operator of Equation (1) and the general spaces $H(V,\chi)$ and $\mathcal{H}(E,\phi)$. Following the scheme a straightforward calculation shows the following:

Lemma 8 *The graph Laplacian $\Delta_{\mathrm{norm}} = d^* d$ with the difference operator d from Equation* (1) *can be explicitly written as*

$$(\Delta^{(n)} f)(l) = \frac{1}{n\chi(d_l)\sqrt{d_l}}\left[\frac{f(l)}{\sqrt{d_l}}\frac{1}{n}\sum_{i=1}^{n}\gamma^2(w_{il})\phi(w_{il}) - \frac{1}{n}\sum_{i=1}^{n}\frac{f(i)}{\sqrt{d_i}}\gamma^2(w_{il})\phi(w_{il})\right].$$

The choice $\chi(d_l) = 1$ and $\gamma^2(w_{il})\phi(w_{il}) = w_{il}$ leads then to the graph Laplacian proposed in Chung and Langlands (1996); Zhou et al. (2004),

$$(\Delta^{(n)} f)(l) = \frac{1}{n\sqrt{d_l}}\left[\frac{f(l)}{\sqrt{d_l}}d_l - \frac{1}{n}\sum_{i=1}^{n}\frac{f(i)}{\sqrt{d_i}}w_{li}\right] = \frac{1}{n}\left[f(l) - \frac{1}{n}\sum_{i=1}^{n} f(i)\frac{w_{il}}{\sqrt{d_l d_i}}\right],$$

or equivalently

$$\Delta^{(n)} f = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}} f = (\mathbb{1} - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}) f.$$

Note that $\Delta^{(\mathrm{u})} = D\Delta^{(\mathrm{rw})}$ and $\Delta^{(\mathrm{n})} = D^{-\frac{1}{2}}\Delta^{(\mathrm{u})}D^{-\frac{1}{2}}$.

3. Limit of the Graph Laplacian for Random Neighborhood Graphs

Before we state the convergence results for the three graph Laplacians on random neighborhood graphs, we first have to define the limit operator. Maybe not surprisingly, in general the Laplace-Beltrami operator will *not* be the limit operator of the graph Laplacian. Instead it will converge to the weighted Laplace-Beltrami operator which is the natural generalization of the Laplace-Beltrami operator for a Riemannian manifold equipped with a non-uniform probability measure. The definition of this limit operator and a discussion of its use for different applications in clustering, semi-supervised learning and regression is the topic of the next section, followed by a sketch of the convergence results.

3.1 Construction of the Neighborhood Graph

We assume to have a sample X_i, $i = 1, \dots, n$ drawn i.i.d. from a probability measure P which has support on a submanifold M. For the exact assumptions regarding P, M and the kernel function k used to define the weights we refer to Section 5.2. The sample then determines the set of vertices V of the graph. Additionally we are given a certain kernel function $k : \mathbb{R}_+ \to \mathbb{R}_+$ and the neighborhood parameter $h \in \mathbb{R}_+^*$. As proposed by Lafon (2004) and Coifman and Lafon (2006), we use this kernel function k to define the following family of data-dependent kernel functions $\tilde{k}_{\lambda,h}$ parameterized by $\lambda \in \mathbb{R}$ as:

$$\tilde{k}_{\lambda,h}(X_i, X_j) = \frac{1}{h^m} \frac{k(\left\|X_i - X_j\right\|^2 / h^2)}{[d_{h,n}(X_i) d_{h,n}(X_j)]^\lambda},$$

where $d_{h,n}(X_i) = \frac{1}{n} \sum_{i=1}^n \frac{1}{h^m} k(\left\|X_i - X_j\right\|^2 / h^2)$ is the degree function introduced in Section 2 with respect to the edge-weights $\frac{1}{h^m} k(\left\|X_i - X_j\right\|^2 / h^2)$. Finally we use $\tilde{k}_{\lambda,h}$ to define the weight $w_{ij} = w(X_i, X_j)$ of the edge between the points X_i and X_j as

$$w_{\lambda,h}(X_i, X_j) = \tilde{k}_{\lambda,h}(X_i, X_j).$$

Note that the case $\lambda = 0$ corresponds to weights with no data-dependent modification. The parameter $h \in \mathbb{R}_+^*$ determines the neighborhood of a point since we will assume that k has compact support, that is X_i and X_j have an edge if $\left\|X_i - X_j\right\| \leq h R_k$ where R_k is the support of kernel function. Note that we will have $k(0) = 0$, so that there are no loops in the graph. This assumption is not necessary, but it simplifies the proofs and makes some of the estimators unbiased.

In Section 2.4 we introduced the random walk, the unnormalized and the normalized graph Laplacian. From now on we consider these graph Laplacians for the random neighborhood graph, that is the weights of the graph w_{ij} have the form $w_{ij} = w(X_i, X_j) = \tilde{k}_{\lambda,h}(X_i, X_j)$. Using the kernel function we can easily extend the graph Laplacians to the whole submanifold M. These extensions can be seen as estimators for the Laplacian on M. We introduce also the extended degree function $\tilde{d}_{\lambda,h,n}$ and the average operator $\tilde{A}_{\lambda,h,n}$,

$$\tilde{d}_{\lambda,h,n}(x) = \frac{1}{n} \sum_{j=1}^n \tilde{k}_{\lambda,h}(x, X_j), \qquad (\tilde{A}_{\lambda,h,n} f)(x) = \frac{1}{n} \sum_{j=1}^n \tilde{k}_{\lambda,h}(x, X_j) f(X_j).$$

Note that $\tilde{d}_{\lambda,h,n} = \tilde{A}_{\lambda,h,n}1$. The extended graph Laplacians are defined as follows:

$$\begin{aligned}
&\text{random walk} && (\Delta^{(\mathrm{rw})}_{\lambda,h,n}f)(x) = \frac{1}{h^2}\Big(f - \tfrac{1}{\tilde{d}_{\lambda,h,n}}\tilde{A}_{\lambda,h,n}f\Big)(x) = \frac{1}{h^2}\left(\frac{\tilde{A}_{\lambda,h,n}g}{\tilde{d}_{\lambda,h,n}}\right)(x), && (3)\\
&\text{unnormalized} && (\Delta^{(\mathrm{u})}_{\lambda,h,n}f)(x) = \frac{1}{h^2}\Big(\tilde{d}_{\lambda,h,n}f - \tilde{A}_{\lambda,h,n}f\Big)(x) = \frac{1}{h^2}(\tilde{A}_{\lambda,h,n}g)(x),\\
&\text{normalized} && (\Delta^{(\mathrm{n})}_{\lambda,h,n}f)(x) = \tfrac{1}{h^2\sqrt{\tilde{d}_{\lambda,h,n}(x)}}\left(\tilde{d}_{\lambda,h,n}\tfrac{f}{\sqrt{\tilde{d}_{\lambda,h,n}}} - \Big(\tilde{A}_{\lambda,h,n}\tfrac{f}{\sqrt{\tilde{d}_{\lambda,h,n}}}\Big)\right)(x)\\
& && \qquad = \tfrac{1}{h^2\sqrt{\tilde{d}_{\lambda,h,n}(x)}}(\tilde{A}_{\lambda,h,n}g')(x),
\end{aligned}$$

where we have introduced $g(y) := f(x) - f(y)$ and $g'(y) := \frac{f(x)}{\sqrt{\tilde{d}_{\lambda,h,n}(x)}} - \frac{f(y)}{\sqrt{\tilde{d}_{\lambda,h,n}(y)}}$. Note that all extensions reproduce the graph Laplacian on the sample:

$$(\Delta f)(i) = (\Delta f)(X_i) = (\Delta_{\lambda,h,n}f)(X_i), \quad \forall i = 1,\ldots,n.$$

The factor $1/h^2$ arises by introducing a factor $1/h$ in the weight γ of the derivative operator d of the graph. This is necessary since d is supposed to approximate a derivative. Since the Laplacian corresponds to a second derivative we get from the definition of the graph Laplacian a factor $1/h^2$.

We would like to note that in the case of the random walk and and the normalized graph Laplacian the normalization with $1/h^m$ in the weights cancels out, whereas it does not cancel for the unnormalized graph Laplacian except in the case $\lambda = 1/2$. The problem here is that in general the intrinsic dimension m of the manifold is unknown. Therefore a normalization with the correct factor $\frac{1}{h^m}$ is not possible, and in the limit $h \to 0$ the estimate of the unnormalized graph Laplacian will generally either vanish or blow up. The easy way to circumvent this is just to rescale the whole estimate such that $\frac{1}{n}\sum_{i=1}^n \tilde{d}_{\lambda,h,n}(X_i)$ equals a fixed constant for every n. The disadvantage is that this method of rescaling introduces a global factor in the limit. A more elegant way might be to simultaneously estimate the dimension m of the submanifold and use the estimated dimension to calculate the correct normalization factor, see, for example, Hein and Audibert (2005). However, in this work we assume for simplicity that for the unnormalized graph Laplacian the intrinsic dimension m of the submanifold is known. It might be interesting to consider both estimates simultaneously, but we leave this as an open problem.

We will consider in the following the limit $h \to 0$, that is the neighborhood of each point X_i shrinks to zero. However, since $n \to \infty$ and h as a function of n approaches zero sufficiently slow, the number of points in each neighborhood approaches ∞, so that roughly spoken sums approximate the corresponding integrals. This is the basic principle behind our convergence result and is well known in the framework of nonparametric regression (see Györfi et al., 2004).

3.2 The Weighted Laplacian and the Continuous Smoothness Functional

The Laplacian is one of the most prominent operators in mathematics. The following general properties are taken from the books of Rosenberg (1997) and Bérard (1986). It occurs in many partial differential equations governing physics, mainly because in Euclidean space it is the only linear second-order differential operator which is translation and rotation invariant. In Euclidean space $\mathbb{R}^d$ it is defined as $\Delta_{\mathbb{R}^d} f = \mathrm{div}(\mathrm{grad} f) = \sum_{i=1}^d \partial_i^2 f$. Moreover, for any domain $\Omega \subseteq \mathbb{R}^d$ it is a negative-semidefinite symmetric operator on $C_c^\infty(\Omega)$, which is a dense subset of $L_2(\Omega)$ (formally self-adjoint), and satisfies

$$\int_\Omega f \Delta h \, dx = - \int_\Omega \langle \nabla f, \nabla h \rangle \, dx.$$

It can be extended to a self-adjoint operator on $L_2(\Omega)$ in several ways depending on the choice of boundary conditions. For any compact domain Ω (with suitable boundary conditions) it can be shown that Δ has a pure point spectrum and the eigenfunctions are smooth and form a complete orthonormal basis of $L_2(\Omega)$, see, for example, Bérard (1986).

The Laplace-Beltrami operator on a Riemannian manifold M is the natural equivalent of the Laplacian in $\mathbb{R}^d$, defined as

$$\Delta_M f = \operatorname{div}(\operatorname{grad} f) = \nabla^a \nabla_a f.$$

However, the more natural definition is the following. For any $f, g \in C_c^\infty(M)$, we have

$$\int_M f \Delta h \, dV(x) = - \int_M \langle \nabla f, \nabla h \rangle \, dV(x),$$

where $dV = \sqrt{\det g}\, dx$ is the natural volume element of M. This definition allows easily an extension to the case where we have a Riemannian manifold M with a measure P . In this paper P will be the probability measure generating the data. We assume in the following that P is absolutely continuous wrt the natural volume element dV of the manifold. Its density is denoted by p. Note that the case when the probability measure is absolutely continuous wrt the Lebesgue measure on $\mathbb{R}^d$ is a special case of our setting.

A recent review article about the weighted Laplace-Beltrami operator is Grigoryan (2006).

Definition 9 (Weighted Laplacian) *Let (M, g_{ab}) be a Riemannian manifold with measure P where P has a differentiable and positive density p with respect to the natural volume element $dV = \sqrt{\det g}\, dx$, and let Δ_M be the Laplace-Beltrami operator on M. For $s \in \mathbb{R}$, we define the s-th weighted Laplacian Δ_s as*

$$\Delta_s := \Delta_M + \frac{s}{p} g^{ab} (\nabla_a p) \nabla_b = \frac{1}{p^s} g^{ab} \nabla_a (p^s \nabla_b) = \frac{1}{p^s} \operatorname{div}(p^s \operatorname{grad}).$$

This definition is motivated by the following equality, for $f, g \in C_c^\infty(M)$,

$$\int_M f(\Delta_s g)\, p^s dV = \int_M f\big(\Delta g + \frac{s}{p} \langle \nabla p, \nabla g \rangle\big) p^s dV = - \int_M \langle \nabla f, \nabla g \rangle \, p^s dV, \tag{4}$$

The family of weighted Laplacians contains two cases which are particularly interesting. The first one, $s = 0$, corresponds to the standard Laplace-Beltrami operator. This case is interesting if one only wants to use properties of the geometry of the manifold but not of the data generating probability measure. The second case, $s = 1$, corresponds to the standard weighted Laplacian $\Delta_1 = \frac{1}{p} \nabla^a (p \nabla_a)$.

In the next section it will turn out that through a data-dependent change of the weights of the graph we can get the just defined weighted Laplacians as the limit operators of the graph Laplacian. The rest of this section will be used to motivate the importance of the understanding of this limit in different applications. Three different but closely related properties of the Laplacian are used in machine learning

- The Laplacian generates the diffusion process. In semi-supervised learning algorithms with a small number of labeled points one would like to propagate the labels along regions of high-density, see Zhu and Ghahramani (2002); Zhu et al. (2003). The limit operator Δ_s shows the influence of a non-uniform density p. The second term $\frac{s}{p}\langle\nabla p, \nabla f\rangle$ leads to an anisotropy in the diffusion. If $s < 0$ this term enforces diffusion in the direction of the maximum of the density whereas diffusion in the direction away from the maximum of the density is weakened. If $s < 0$ this is just the other way round.

- The smoothness functional induced by the weighted Laplacian Δ_s, see Equation (4), is given by
$$S(f) = \int_M \|\nabla f\|^2 \, p^s \, dV.$$
For $s > 0$ this smoothness functional prefers functions which are smooth in high-density regions whereas unsmooth behavior in low-density is penalized less. This property can also be interesting in semi-supervised learning where one assumes especially when only a few labeled points are known that the classifier should be constant in high-density regions whereas changes of the classifier are allowed in low-density regions, see Bousquet et al. (2004) for some discussion of this point and Hein (2005, 2006) for a proof of convergence of the regularizer induced by the graph Laplacian towards the smoothness functional $S(f)$. In Figure 1 this is illustrated by mapping a density profile in $\mathbb{R}^2$ onto a two-dimensional manifold. However,

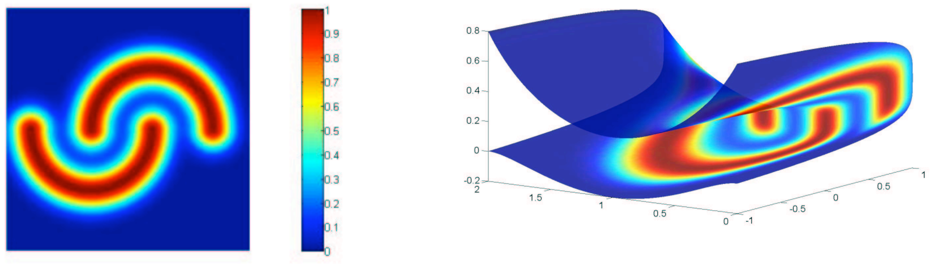


Figure 1: A density profile mapped onto a two-dim. submanifold in $\mathbb{R}^3$ with two clusters.

also the case $s < 0$ can be interesting. Minimizing the smoothness functional $S(f)$ implies that one enforces smoothness of the function f where one has little data, and one allows the function to vary more where one has sampled a lot of data points. Such a penalization has been considered by Canu and Elisseeff (1999) for regression.

- The eigenfunctions of the Laplacian Δ_s can be seen as the limit partioning of spectral clustering for the normalized graph Laplacian (however, a rigorous mathematical proof has not been given yet, see von Luxburg et al., 2007, for a convergence result for fixed h). If $s = 0$ one gets just a geometric clustering in the sense that irrespectively of the probability measure generating the data the clustering is determined by the geometry of the submanifold. If $s > 0$

the eigenfunction corresponding to the first non-zero eigenvalue is likely to change its sign in a low-density region. This argument follows from the previous discussion on the smoothness functional $S(f)$ and the Rayleigh-Ritz principle. Let us assume for a moment that M is compact without boundary and that $p(x) > 0, \forall x \in M$, then the eigenspace corresponding to the first eigenvalue $\lambda_0 = 0$ is given by the constant functions. The first non-zero eigenvalue can then be determined by the Rayleigh-Ritz variational principle

$$\lambda_1 = \inf_{u \in C^\infty(M)} \left\{ \frac{\int_M \|\nabla u\|^2 p(x)^s dV(x)}{\int_M u^2(x) p(x)^s dV(x)} \;\middle|\; \int_M u(x)\, p(x)^s dV(x) = 0 \right\}.$$

Since the first eigenfunction has to be orthogonal to the constant functions, it has to change its sign. However, since $\|\nabla u\|^2$ is weighted by a power of the density p^s it is obvious that for $s > 0$ the function will change its sign in a region of low density.

3.3 Limit of the Graph Laplacians

The following theorem summarizes and slightly weakens the results of Theorem 30 and Theorem 31 of Section 5.

Main Result *Let M be a m-dimensional submanifold in $\mathbb{R}^d$, $\{X_i\}_{i=1}^n$ a sample from a probability measure P on M with density p. Let $x \in M \backslash \partial M$ and define $s = 2(1-\lambda)$. Then under technical conditions on the submanifold M, the kernel k and the density p introduced in Section 5, if $h \to 0$ and $nh^{m+2}/\log n \to \infty$,*

$$\begin{array}{lll} \text{random walk:} & \lim_{n\to\infty} (\Delta^{(\mathrm{rw})}_{\lambda,h,n} f)(x) \;\sim\; -(\Delta_s f)(x) & \text{almost surely,} \\ \text{unnormalized:} & \lim_{n\to\infty} (\Delta^{(\mathrm{u})}_{\lambda,h,n} f)(x) \;\sim\; -p(x)^{1-2\lambda} (\Delta_s f)(x) & \text{almost surely.} \end{array}$$

The optimal rate is obtained for $h(n) = O\Big((\log n/n)^{\frac{1}{m+4}}\Big)$. If $h \to 0$ and $nh^{m+4}/\log n \to \infty$,

$$\text{normalized:} \quad \lim_{n\to\infty} (\Delta^{(\mathrm{n})}_{\lambda,h,n} f)(x) \;\sim\; -p(x)^{\frac{1}{2}-\lambda} \Delta_s \left(\frac{f}{p^{\frac{1}{2}-\lambda}} \right)(x) \quad \text{almost surely.}$$

where $\sim$ means that there exists a constant only depending on the kernel k and λ such that equality holds.

The first observation is that the conjecture that the graph Laplacian approximates the Laplace-Beltrami operator is only true for the uniform measure, where p is constant. In this case all limits agree up to constants. However, big differences arise when one has a non-uniform measure on the submanifold, which is the generic case in machine learning applications. In this case all limits disagree and only the random walk graph Laplacian converges towards the weighted Laplace-Beltrami operator which is the natural generalization of the Laplace-Beltrami operator when the manifold is equipped with a non-uniform probability measure. The unnormalized graph Laplacian has the additional factor $p^{1-2\lambda}$. However, this limit is actually quite useful, when one thinks of applications of so called label propagation algorithms in semi-supervised learning. If one uses this graph Laplacian as the diffusion operator to propagate the labeled data, it means that the diffusion for $\lambda < 1/2$ is faster in regions where the density is high. The consequence is that labels in regions of high density are propagated faster than labels in low-density regions. This makes sense since under the cluster assumption labels in regions of low density are less informative than labels in regions of high density. In general, from the viewpoint of a diffusion process the weighted Laplace-Beltrami operator $\Delta_s = \Delta_M + \frac{s}{p}\nabla p \nabla$ can be seen as inducing an anisotropic diffusion due to the extra term $\frac{s}{p}\nabla p \nabla$, which is directed towards or away from increasing density depending on s. This is a desired property in semi-supervised learning, where one actually wants that the diffusion is mainly along regions of the same density level in order to fulfill the cluster assumption.

The second observation is that the data-dependent modification of edge weights allows to control the influence of the density on the limit operator as observed by Coifman and Lafon (2006). In fact one can even eliminate it for $s=0$ resp. $\lambda=1$ in the case of the random walk graph Laplacian. This could be interesting in computer graphics where the random walk graph Laplacian is used for mesh and point cloud processing, see, for example, Sorkine (2006). If one has gathered points of a curved object with a laser scanner it is likely that the points have a non-uniform distribution on the object. Its surface is a two-dimensional submanifold in $\mathbb{R}^3$. In computer graphics the non-uniform measure is only an artefact of the sampling procedure and one is only interested in the Laplace-Beltrami operator to infer geometric properties. Therefore the elimination of the influence of a non-uniform measure on the submanifold is of high interest there. We note that up to a multiplication with the inverse of the density the elimination of density effects is also possible for the unnormalized graph Laplacian, but not for the normalized graph Laplacian. All previous observations are naturally also true if the data does not lie on a submanifold but has d-dimensional support in $\mathbb{R}^d$.

The interpretation of the limit of the normalized graph Laplacian is more involved. An expansion of the limit operator shows the complex dependency on the density p:

$$p^{\frac{1}{2}-\lambda}\Delta_s\left(\frac{f}{p^{\frac{1}{2}-\lambda}}\right) = \Delta_M f + \frac{1}{p}\nabla p \nabla f - (\lambda - \frac{1}{2})^2 \frac{f}{p}\left\|\nabla p\right\|^2 + (\lambda - \frac{1}{2})\frac{f}{p}\Delta_M p.$$

We leave it to the reader to think of possible applications of this Laplacian.

The discussion shows that the choice of the graph Laplacian depends on what kind of problem one wants to solve. Therefore, in our opinion there is no universal best choice between the random walk and the unnormalized graph Laplacian from a machine learning point of view. However, from a mathematical point of view only the random walk graph Laplacian has the correct (pointwise) limit to the weighted Laplace-Beltrami operator.

4. Illustration of the Results

In this section we want to illustrate the differences between the three graph Laplacians and the control of the influence of the data-generating measure via the parameter λ.

4.1 Flat Space $\mathbb{R}^2$

In the first example the data lies in Euclidean space $\mathbb{R}^2$. Here we want to show two things. First, the sketch of the main result shows that if the data generating measure is uniform all graph Laplacians converge for all values of the reweighting parameter λ up to constants to the Laplace-Beltrami operator, which is in the case of $\mathbb{R}^2$ just the standard Laplacian. In Figure 2 the estimates of the three graph Laplacians are shown for the uniform measure $[-3,3]^2$ and $\lambda=0$. It can be seen that up to a scaling all estimates agree very well. In a second example we study the effect of a non-uniform data-generating measure. In general all estimates disagree in this case. We illustrate this effect

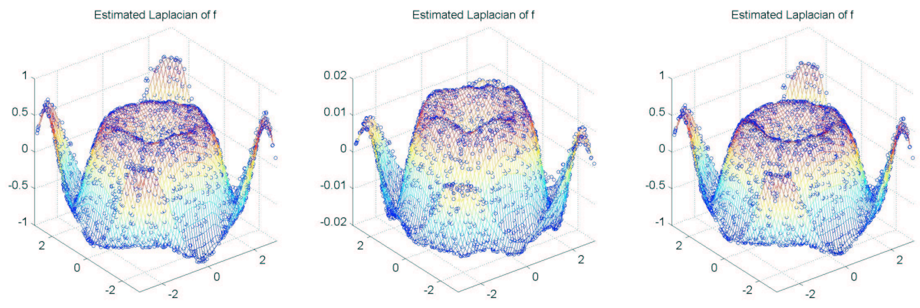


Figure 2: For the uniform distribution all graph Laplacians, $\Delta^{(\mathrm{rw})}_{\lambda,h,n}$, $\Delta^{(\mathrm{u})}_{\lambda,h,n}$ and $\Delta^{(\mathrm{n})}_{\lambda,h,n}$ (from left to right) agree up to constants for all λ. In the figure the estimates of the Laplacian are shown for the uniform measure on $[-3,3]^2$ and the function $f(x) = \sin(\frac{1}{2}\|x\|^2)/\|x\|^2$ with 2500 samples and $h = 1.4$.

in the case of $\mathbb{R}^2$ with a Gaussian distribution $\mathcal{N}(0,1)$ as data-generating measure and the simple function $f(x) = \sum_i x_i - 4$. Note that $\Delta f = 0$ so that for the random walk and the unnormalized graph Laplacian only the anisotropic part of the limit operator, $\frac{1}{p}\nabla p \nabla f$ is non-zero. Explicitly the limits are given as

$$\begin{aligned}
\Delta^{(\mathrm{rw})} &\sim & \Delta_s f &= \Delta f + \frac{s}{p}\nabla p \nabla f = -s\sum\nolimits_i x_i, \\
\Delta^{(\mathrm{u})} &\sim & p^{1-2\lambda}\Delta_s f &= -s\, e^{-\frac{1-2\lambda}{2}\|x\|^2}\sum\nolimits_i x_i, \\
\Delta^{(\mathrm{n})} &\sim & p^{\frac{1}{2}-\lambda}\Delta_s \frac{f}{p^{\frac{1}{2}-\lambda}} &= -\sum\nolimits_i x_i - \big(\sum\nolimits_i x_i - 4\big)\Big[(\lambda-\frac{1}{2})(\frac{3}{2}-\lambda)\|x\|^2 - 2(\lambda-\frac{1}{2})\Big].
\end{aligned}$$

This shows that even applied to simple functions there can be large differences between the different limit operators provided the samples come from a non-uniform probability measure. Note that like in nonparametric kernel regression the estimate is quite bad at the boundary. This well known boundary effect arises since at the boundary one does not average over a full ball but only over some part of a ball. Thus the first derivative ∇f of order $O(h)$ does not cancel out so that multiplied with the factor $1/h^2$ we have a term of order $O(1/h)$ which blows up. Roughly spoken this effect takes place at all points of order $O(h)$ away from the boundary, see also Coifman and Lafon (2006).

4.2 The Sphere S^2

In our next example we consider the case where the data lies on a submanifold M in $\mathbb{R}^d$. Here we want to illustrate in the case of a sphere S^2 in $\mathbb{R}^3$ the control of the influence of the density via the parameter λ. In this case we sample from the probability measure with density $p(\phi,\theta) = \frac{1}{8\pi} + \frac{3}{8\pi}\cos^2(\theta)$ in spherical coordinates with respect to the volume element $dV = \sin(\theta)d\theta d\phi$. This density has a two-cluster structure on the sphere, where the northern and southern hemisphere represent one cluster. An estimate of the density p is shown in the Figure 4. We show the results of the

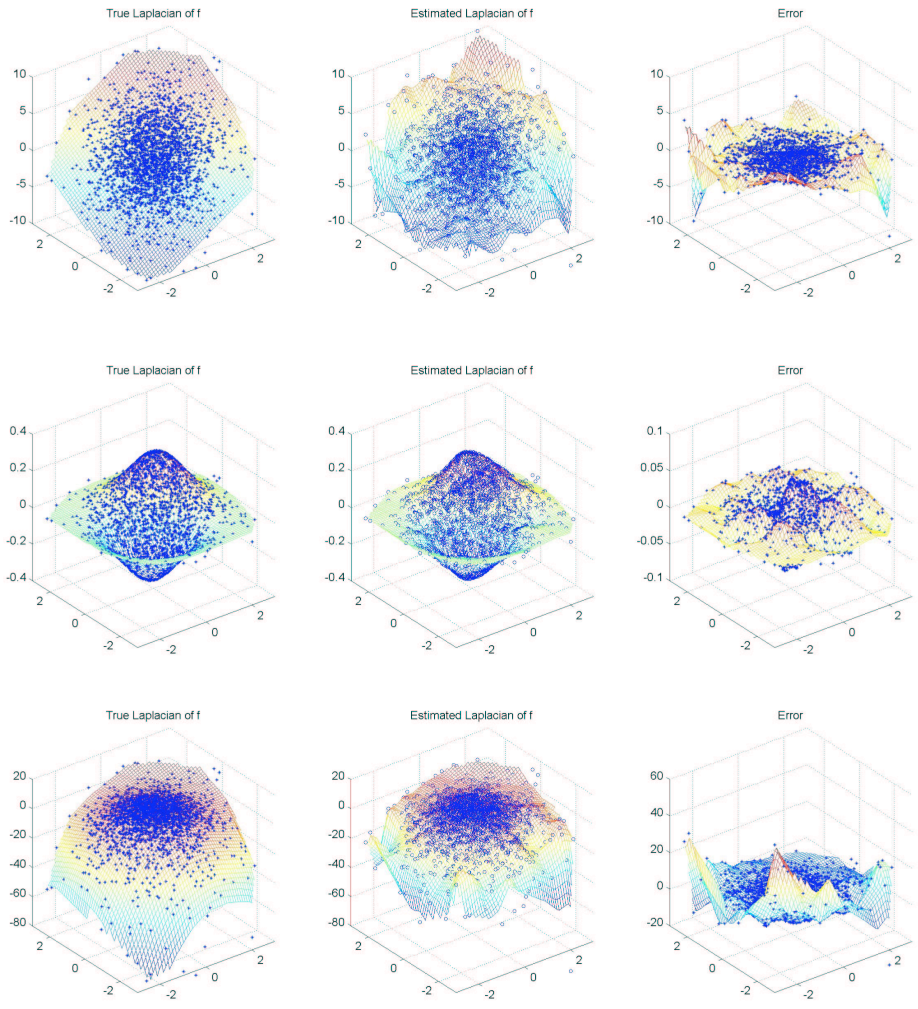


Figure 3: Illustration of the differences of the three graph Laplacians, random walk, unnormalized and normalized (from the top) for $\lambda = 0$. The function f is $f = \sum_{i=1}^{2} x_i - 4$ and the 2500 samples come from a standard Gaussian distribution on $\mathbb{R}^2$. The neighborhood size h is set to 1.2.

random walk graph Laplacian together with the result of the weighted Laplace-Beltrami operator and an error plot for $\lambda = 0, 1, 2$ resulting in $s = -2, 0, 2$ for the function $f(\phi, \theta) = \cos(\theta)$. First one can see that for a non-uniform probability measure the results for different values of λ differ quite a lot. Note that the function f is adapted to the cluster structure in the sense that it does not change much in each cluster but changes very much in region of low density. In the case of $s = 2$ we can see that $\Delta_s f$ would lead to a diffusion which would lead roughly to a kind of step function which changes at the equator. The same is true for $s = 0$ but the effect is much smaller than for $s = 2$. In the case of $s = -2$ we have a completely different behavior. $\Delta_s f$ has now flipped its sign near to the equator so that the induced diffusion process would try to smooth the function in the low density region.

5. Proof of the Main Result

In this section we will present the main results which were sketched in Section 3.3 together with the proofs. In Section 5.1 we first introduce some non-standard tools from differential geometry which we will use later on. In particular, it turns out that the so called manifolds with boundary of bounded geometry are the natural framework where one can still deal with non-compact manifolds in a setting comparable to the compact case. After a proper statement of the assumptions under which we prove the convergence results of the graph Laplacian and a preliminary result about convolutions on submanifolds which is of interest on its own, we then start with the final proofs. The proof is basically divided into two parts, the bias and the variance, where these terms are only approximately valid. The reader not familiar with differential geometry is encouraged to first read the appendix on basics of differential geometry in order to be equipped with the necessary background.

5.1 Non-compact Submanifolds in $\mathbb{R}^d$ with Boundary

We prove the pointwise convergence for non-compact submanifolds. Therefore we have to restrict the class of submanifolds since manifolds with unbounded curvature do not allow reasonable function spaces.

Remark 10 *In the rest of this paper we use the Einstein summation convention that is over indices occurring twice has to be summed. Note that the definition of the curvature tensor differs between textbooks. We use here the conventions regarding the definitions of curvature etc. of Lee (1997).*

5.1.1 Manifolds with Boundary of Bounded Geometry

We will consider in general non-compact submanifolds with boundary. In textbooks on Riemannian geometry one usually only finds material for the case where the manifold has no boundary. Also the analysis, for example the definition of Sobolev spaces on non-compact Riemannian manifolds, seems to be non-standard. We profit here very much from the thesis and an accompanying paper of Schick (1996, 2001) which introduces manifolds with boundary of bounded geometry. All material of this section is taken from these articles. Naturally this plus of generality leads also to a slightly larger technical overload. Nevertheless we think that it is worth this effort since the class of manifolds with boundary of bounded geometry includes almost any kind of submanifold one could have in mind. Moreover, to our knowledge, it is the most general setting where one can still introduce a notion of Sobolev spaces with the usual properties.

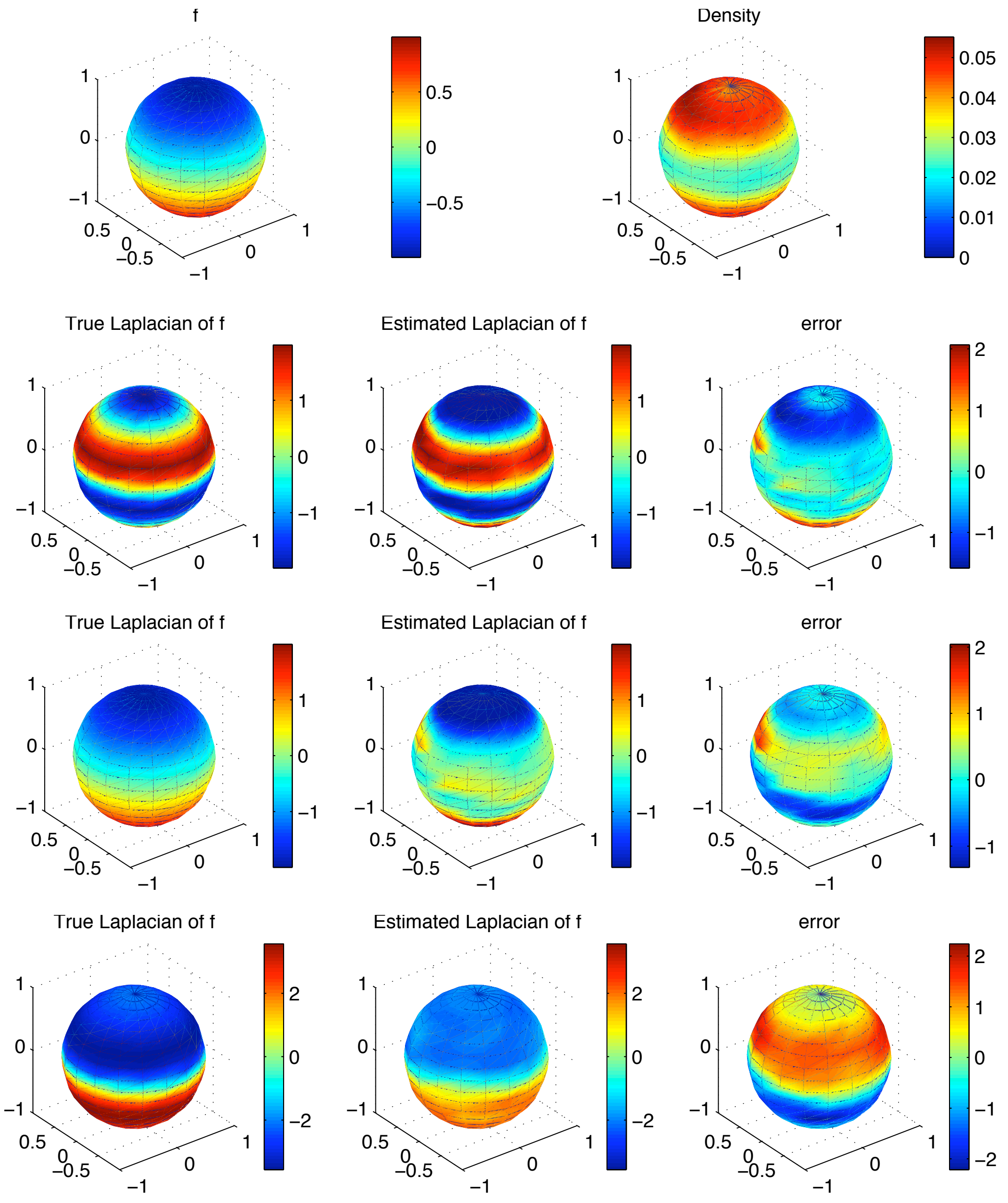


Figure 4: Illustration of the effect of $\lambda = 0, 1, 2$ (row $2-4$) resulting in $s = -2, 0, 2$ for the sphere with a non-uniform data-generating probability measure and the function $f(\theta, \phi) = \cos(\theta)$ (row 1) for the random walk Laplacian with $n = 2500$ and $h = 0.6$

Note that the boundary ∂M is an isometric submanifold of M of dimension $m-1$. Therefore it has a second fundamental form $\overline{\Pi}$ which should not be mixed up with the second fundamental form Π of M which is with respect to the ambient space $\mathbb{R}^d$. We denote by $\overline{\nabla}$ the connection and by $\overline{R}$ the curvature of ∂M. Moreover, let ν be the normal inward vector field at ∂M.

Definition 11 (Manifold with boundary of bounded geometry) *Let M be a manifold with boundary ∂M (possibly empty). It is of bounded geometry if the following holds:*

- *(N) Normal Collar: there exists $r_C > 0$ so that the normal geodesic flow,*
$$K : (x,t) \to \exp_x(t\nu_x),$$
is defined on $\partial M \times [0, r_C)$ and is a diffeomorphism onto its image (ν_x is the inward normal vector). Let $N(s) := K(\partial M \times [0,s])$ be the collar set for $0 \le s \le r_C$.

- *(IC) The injectivity radius $\mathrm{inj}_{\partial M}$ of ∂M is positive.*

- *(I) Injectivity radius of M: There is $r_i > 0$ so that if $r \le r_i$ then for $x \in M \backslash N(r)$ the exponential map is a diffeomorphism on $B_M(0,r) \subset T_xM$ so that normal coordinates are defined on every ball $B_M(x,r)$ for $x \in M \backslash N(r)$.*

- *(B) Curvature bounds: For every $k \in \mathbb{N}$ there is C_k so that $|\nabla^i R| \le C_k$ and $\overline{\nabla}^i \overline{\Pi} \le C_k$ for $0 \le i \le k$, where ∇^i denotes the covariant derivative of order i.*

Note that (B) imposes bounds on all orders of the derivatives of the curvatures. One could also restrict the definition to the order of derivatives needed for the goals one pursues. But this would require even more notational effort, therefore we skip this. In particular, in Schick (1996) it is argued that boundedness of all derivatives of the curvature is very close to the boundedness of the curvature alone.

The lower bound on the injectivity radius of M and the bound on the curvature are standard to define manifolds of bounded geometry without boundary. Now the problem of the injectivity radius of M is that at the boundary it somehow makes only partially sense since $\mathrm{inj}_M(x) \to 0$ as $d(x, \partial M) \to 0$. Therefore one replaces next to the boundary standard normal coordinates with normal collar coordinates.

Definition 12 (normal collar coordinates) *Let M be a Riemannian manifold with boundary ∂M. Fix $x' \in \partial M$ and an orthonormal basis of $T_{x'}\partial M$ to identify $T_{x'}\partial M$ with $\mathbb{R}^{m-1}$. For $r_1, r_2 > 0$ sufficiently small (such that the following map is injective) define normal collar coordinates,*
$$n_{x'} : B_{\mathbb{R}^{m-1}}(0, r_1) \times [0, r_2] \to M : (v,t) \to \exp^M_{\exp^{\partial M}_{x'}(v)}(t\nu).$$
The pair (r_1, r_2) is called the width of the normal collar chart $n_{x'}$.

The next proposition shows why manifolds of bounded geometry are especially interesting.

Proposition 13 (Schick 2001) *Assume that conditions $(N), (IC), (I)$ of Definition 11 hold.*

- *(B1) There exist* $0 < R_1 \leq r_{\mathrm{inj}}(\partial M)$, $0 < R_2 \leq r_C$ *and* $0 < R_3 \leq r_i$ *and constants* $C_K > 0$ *for each* $K \in \mathbb{N}$ *such that whenever we have normal boundary coordinates of width* (r_1, r_2) *with* $r_1 \leq R_1$ *and* $r_2 \leq R_2$ *or normal coordinates of radius* $r_3 \leq r_i$ *then in these coordinates,*

$$|D^\alpha g_{ij}| \leq C_K \quad \text{and} \quad |D^\alpha g^{ij}| \leq C_K \quad \text{for all} \quad |\alpha| \leq K.$$

The condition (B) *in Definition 11 holds if and only if* $(B1)$ *holds. The constants* C_K *can be chosen to depend only on* $r_i, r_C, \mathrm{inj}_{\partial M}$ *and* C_k*.*

Note that due to $g^{ij} g_{jk} = \delta^i_k$ one gets upper and lower bounds on the operator norms of g and g^{-1}, respectively, which result in upper and lower bounds for $\sqrt{\det g}$. This implies that we have upper and lower bounds on the volume form $dV(x) = \sqrt{\det g}\, dx$.

Lemma 14 (Schick 2001) *Let* (M, g) *be a Riemannian manifold with boundary of bounded geometry of dimension m. Then there exists* $R_0 > 0$ *and constants* $S_1 > 0$ *and* S_2 *such that for all* $x \in M$ *and* $r \leq R_0$ *one has*

$$S_1 r^m \leq \mathrm{vol}(B_M(x,r)) \leq S_2 r^m.$$

Another important tool for analysis on manifolds are appropriate function spaces. In order to define a Sobolev norm one first has to fix a family of charts U_i with $M \subset \cup_i U_i$ and then define the Sobolev norm with respect to these charts. The resulting norm will depend on the choice of the charts U_i. Since in differential geometry the choice of the charts should not matter, the natural question arises how the Sobolev norm corresponding to a different choice of charts V_i is related to that for the choice U_i. In general, the Sobolev norms will not be the same. However, if one assumes that the transition maps are smooth and the manifold M is compact then the resulting norms will be equivalent and therefore define the same topology. Now if one has a non-compact manifold this argumentation does not work anymore. This problem is solved in general by defining the norm with respect to a covering of M by normal coordinate charts. Then it can be shown that the change of coordinates between these normal coordinate charts is well-behaved due to the bounded geometry of M. In that way it is possible to establish a well-defined notion of Sobolev spaces on manifolds with boundary of bounded geometry in the sense that any norm defined with respect to a different covering of M by normal coordinate charts is equivalent. Let $(U_i, \phi_i)_{i \in I}$ be a countable covering of the submanifold M with normal coordinate charts of M, that is $M \subset \cup_{i \in I} U_i$, then:

$$\|f\|_{C^k(M)} = \max_{m \leq k} \sup_{i \in I} \sup_{x \in \phi_i(U_i)} \left| D^m (f \circ \phi_i^{-1})(x) \right|.$$

In the following we will denote with $C^k(M)$ the space of C^k-functions on M together with the norm $\|\cdot\|_{C^k(M)}$.

5.1.2 INTRINSIC VERSUS EXTRINSIC PROPERTIES

Most of the proofs for the continuous part will work with Taylor expansions in normal coordinates. It is then of special interest to have a connection between intrinsic and extrinsic distances. Since the distance on M is induced from $\mathbb{R}^d$, it is obvious that one has $\|x - y\|_{\mathbb{R}^d} \sim d_M(x,y)$ for all $x, y \in M$ which are sufficiently close. The next proposition proven by Smolyanov, von Weizsäcker, and Wittich (2000) provides an asymptotic expression of geometric quantities of the submanifold M in the neighborhood of a point $x \in M$. Particularly, it gives a third-order approximation of the intrinsic distance $d_M(x,y)$ in M in terms of the extrinsic distance in the ambient space X which is in our case just the Euclidean distance in $\mathbb{R}^d$.

Proposition 15 *Let $i : M \to \mathbb{R}^d$ be an isometric embedding of the smooth m-dimensional Riemannian manifold M into $\mathbb{R}^d$. Let $x \in M$ and V be a neighborhood of 0 in $\mathbb{R}^m$ and let $\Psi : V \to U$ provide normal coordinates of a neighborhood U of x, that is $\Psi(0) = x$. Then for all $y \in V$:*

$$\|y\|^2_{\mathbb{R}^m} = d_M^2(x, \Psi(y)) = \|(i \circ \Psi)(y) - i(x)\|^2 + \frac{1}{12}\|\Pi(\dot\gamma,\dot\gamma)\|^2_{T_x\mathbb{R}^d} + O(\|y\|^5_{\mathbb{R}^m}),$$

where Π is the second fundamental form of M and γ the unique geodesic from x to $\Psi(y)$ such that $\dot\gamma = y^i \partial_{y^i}$. The volume form $dV = \sqrt{\det g_{ij}(y)}\, dy$ of M satisfies in normal coordinates,

$$dV = \Big(1 + \frac{1}{6} R_{iuvi}\, y^u y^v + O(\|y\|^3_{\mathbb{R}^m})\Big) dy,$$

In particular,

$$(\Delta \sqrt{\det g_{ij}})(0) = -\frac{1}{3} R,$$

where R is the scalar curvature (i.e., $R = g^{ik} g^{jl} R_{ijkl}$).

We would like to note that in Smolyanov, von Weizsäcker, and Wittich (2007) this proposition was formulated for general ambient spaces X, that is arbitrary Riemannian manifolds X. Using the more general form of this proposition one could extend the results in this paper to submanifolds of other ambient spaces X. However, in order to use the scheme one needs to know the geodesic distances in X, which are usually not available for general Riemannian manifolds. Nevertheless, for some special cases like the sphere, one knows the geodesic distances. Submanifolds of the sphere could be of interest, for example in geophysics or astronomy.

The previous proposition is very helpful since it gives an asymptotic expression of the geodesic distance $d_M(x,y)$ on M in terms of the extrinsic Euclidean distance. The following lemma is a non-asymptotic statement taken from Bernstein et al. (2001) which we present in a slightly different form. But first we establish a connection between what they call the 'minimum radius of curvature' and upper bounds on the extrinsic curvatures of M and ∂M. Let

$$\Pi_{\max} = \sup_{x \in M}\ \sup_{v \in T_xM, \|v\|=1} \|\Pi(v,v)\|, \qquad \overline{\Pi}_{\max} = \sup_{x \in \partial M}\ \sup_{v \in T_x\partial M, \|v\|=1} \left\|\overline{\Pi}(v,v)\right\|,$$

where $\overline{\Pi}$ is the second fundamental form of ∂M as a submanifold of M. We set $\overline{\Pi}_{\max} = 0$ if the boundary ∂M is empty.

Using the relation between the acceleration in the ambient space and the second fundamental form for unit-speed curves γ with no acceleration in M ($D_t\dot\gamma = 0$) established in Section A.3, we get for the Euclidean acceleration of such a curve γ in $\mathbb{R}^d$,

$$\|\ddot\gamma\| = \|\Pi(\dot\gamma,\dot\gamma)\|.$$

Now if one has a non-empty boundary ∂M it can happen that a length-minimizing curve goes (partially) along the boundary (imagine $\mathbb{R}^d$ with a ball at the origin cut out). Then the segment c along the boundary will be a geodesic of the submanifold ∂M, see Alexander and Alexander (1981), that is $\overline{D}_t \dot{c} = \overline{\nabla}_{\dot{c}} \dot{c} = 0$ where $\overline{\nabla}$ is the connection of ∂M induced by M. However, c will not be a geodesic in M (in the sense of a curve with no acceleration) since by the Gauss-Formula in Theorem 41,

$$D_t \dot{c} = \overline{D}_t \dot{c} + \overline{\Pi}(\dot{c}, \dot{c}) = \overline{\Pi}(\dot{c}, \dot{c}).$$

Therefore, in general the upper bound on the Euclidean acceleration of a length-minimizing curve γ in M is given by,

$$\|\ddot{\gamma}\| = \left\|\overline{\Pi}(\dot{\gamma}, \dot{\gamma}) + \Pi(\dot{\gamma}, \dot{\gamma})\right\| \le \overline{\Pi}_{\max} + \Pi_{\max}.$$

Using this inequality, one can derive a lower bound on the 'minimum radius of curvature' ρ defined in Bernstein et al. (2001) as $\rho = \inf\{1/\|\ddot{\gamma}\|_{\mathbb{R}^d}\}$ where the infimum is taken over all unit-speed geodesics γ of M (in the sense of length-minimizing curves):

$$\rho \ge \frac{1}{\overline{\Pi}_{\max} + \Pi_{\max}}.$$

Finally we can formulate the Lemma from Bernstein et al. (2001).

Lemma 16 *Let $x, y \in M$ with $d_M(x, y) \le \pi\rho$. Then*

$$2\rho \sin(d_M(x, y)/(2\rho)) \le \|x - y\|_{\mathbb{R}^d} \le d_M(x, y).$$

Noting that $\sin(x) \ge x/2$ for $0 \le x \le \pi/2$, we get as an easier to handle corollary:

Corollary 17 *Let $x, y \in M$ with $d_M(x, y) \le \pi\rho$. Then*

$$\frac{1}{2} d_M(x, y) \le \|x - y\|_{\mathbb{R}^d} \le d_M(x, y).$$

In the given form this corollary is quite useless since we only have the Euclidean distances between points and therefore we have no possibility to check the condition $d_M(x, y) \le \pi\rho$. In general small Euclidean distance does not imply small intrinsic distance. Imagine a circle where one has cut out a very small segment. Then the Euclidean distance between the two ends is very small however the geodesic distance is very large. We show now that under an additional assumption one can transform the above corollary so that one can use it when one has only knowledge about Euclidean distances.

Lemma 18 *Let M have a finite radius of curvature $\rho > 0$. We further assume that,*

$$\kappa := \inf_{x \in M} \inf_{y \in M \setminus B_M(x, \pi\rho)} \|x - y\|,$$

is non-zero. Then $B_{\mathbb{R}^d}(x, \kappa/2) \cap M \subset B_M(x, \kappa) \subset B_M(x, \pi\rho)$. Particularly, if $x, y \in M$ and $\|x - y\| \le \kappa/2$, then

$$\frac{1}{2} d_M(x, y) \le \|x - y\|_{\mathbb{R}^d} \le d_M(x, y) \le \kappa.$$

Proof By definition κ is at most the infimum of $\|x - y\|$ where y satisfies $d_M(x, y) = \pi\rho$. Therefore the set $B_{\mathbb{R}^d}(x, \kappa/2) \cap M$ is a subset of $B_M(x, \pi\rho)$. The rest of the lemma then follows by Corollary 17. Figure 5 illustrates this construction. ■

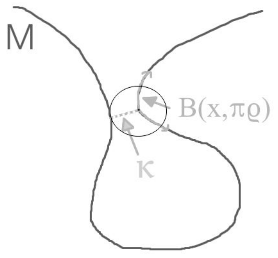


Figure 5: κ is the Euclidean distance of $x \in M$ to $M \backslash B_M(x, \pi\rho)$.

5.2 Notations and Assumptions

In general we work on complete non-compact manifolds with boundary. Compared to a setting where one considers only compact manifolds one needs a slightly larger technical overhead. However, we will indicate how the technical assumptions simplify if one has a compact submanifold with boundary or even a compact manifold without boundary.

We impose the following assumptions on the manifold M:

Assumption 19 *(i) The map $i : M \to \mathbb{R}^d$ is a smooth embedding,*

(ii) The manifold M with the metric induced from $\mathbb{R}^d$ is a smooth manifold with boundary of bounded geometry (possibly $\partial M = \emptyset$),

(iii) M has bounded second fundamental form,

(iv) It holds $\kappa := \inf_{x\in M} \inf_{y\in M\backslash B_M(x,\pi\rho)} \|i(x) - i(y)\| > 0$, where ρ is the radius of curvature defined in Section 5.1.2,

(v) For any $x \in M\backslash\partial M$, $\delta(x) := \inf\limits_{y\in M\backslash B_M(x,\frac{1}{3}\min\{\mathrm{inj}(x),\pi\rho\})} \|i(x) - i(y)\|_{\mathbb{R}^d} > 0$, where $\mathrm{inj}(x)$ is the injectivity radius[1] at x and $\rho > 0$ is the radius of curvature.

The first condition ensures that $i(M)$ is a smooth submanifold of $\mathbb{R}^d$. Usually we do not distinguish between $i(M)$ and M. The use of the abstract manifold M as a starting point emphasizes that there exists an m-dimensional smooth manifold M or roughly equivalent an m-dimensional smooth parameter space underlying the data. The choice of the d features determines then the representation in $\mathbb{R}^d$. The choice of features corresponds therefore to a specific choice of the inclusion map i since i determines how M is embedded into $\mathbb{R}^d$. This means that another choice of features leads in general to a different mapping i but the initial abstract manifold M is always the same. However, in the second condition we assume that the metric structure of M is induced by $\mathbb{R}^d$ (which implies that i is trivially an isometric embedding). Therefore the metric structure depends on the embedding i or equivalently on our choice of features.

The second condition ensures that M is an isometric submanifold of $\mathbb{R}^d$ which is well-behaved. As discussed in Section 5.1.1, manifolds of bounded geometry are in general non-compact, complete Riemannian manifolds with boundary where one has uniform control over all intrinsic curvatures. The uniform bounds on the curvature allow to do reasonable analysis in this general setting. In

1. Note that the injectivity radius $\mathrm{inj}(x)$ is always positive.

particular, it allows us to introduce the function spaces $C^k(M)$ with their associated norm. It might be possible to prove pointwise results even without the assumption of bounded geometry. But we think that the setting studied here is already general enough to encompass all cases encountered in practice.

The third condition ensures that M also has well-behaved extrinsic geometry and implies that the radius of curvature ρ is lower bounded. Together with the fourth condition it enables us to get global upper and lower bounds of the intrinsic distance on M in terms of the extrinsic distance in $\mathbb{R}^d$ and vice versa, see Lemma 18.

The fourth condition is only necessary in the case of non-compact submanifolds. It prevents the manifold from self-approaching. More precisely it ensures that if parts of M are far away from x in the geometry of M they do not come too close to x in the geometry of $\mathbb{R}^d$. Assuming that $i(M)$ is a submanifold, this assumption is already included implicitly. However, for non-compact submanifolds the self-approaching could happen at infinity. Therefore we exclude it explicitly. Moreover, note that for submanifolds with boundary one has $\mathrm{inj}(x) \to 0$ as x approaches the boundary[2] ∂M. Therefore also $\delta(x) \to 0$ as $d(x, \partial M) \to 0$. However, this behavior of $\delta(x)$ at the boundary does not matter for the proof of pointwise convergence in the interior of M.

Note that if M is a smooth and *compact* manifold conditions (ii)-(v) hold automatically.

In order to emphasize the distinction between extrinsic and intrinsic properties of the manifold we always use the slightly cumbersome notations $x \in M$ (intrinsic) and $i(x) \in \mathbb{R}^d$ (extrinsic). The reader who is not familiar with Riemannian geometry should keep in mind that locally, a submanifold of dimension m looks like $\mathbb{R}^m$. This becomes apparent if one uses normal coordinates. Also the following dictionary between terms of the manifold M and the case when one has only an open set in $\mathbb{R}^d$ (i is then the identity mapping) might be useful.

Manifold M	open set in $\mathbb{R}^d$
g_{ij} , $\sqrt{\det g}$	δ_{ij} , 1
natural volume element	Lebesgue measure
Δ_s	$\Delta_s = \sum_{i=1}^d \frac{\partial^2}{\partial (z_i)^2} + \frac{s}{p} \sum_{i=1}^d \frac{\partial p}{\partial z^i} \frac{\partial}{\partial z^i}$

The kernel functions which are used to define the weights of the graph are always functions of the squared norm in $\mathbb{R}^d$. Furthermore, we make the following assumptions on the kernel function k:

Assumption 20 *(i)* $k : \mathbb{R}_+ \to \mathbb{R}$ *is measurable, non-negative and non-increasing on* $\mathbb{R}_+^*$,

(ii) $k \in C^2(\mathbb{R}_+^*)$, *that is in particular* k, $\frac{\partial k}{\partial x}$ *and* $\frac{\partial^2 k}{\partial x^2}$ *are bounded,*

(iii) k, $|\frac{\partial k}{\partial x}|$ *and* $|\frac{\partial^2 k}{\partial x^2}|$ *have exponential decay:* $\exists c, \alpha, A \in \mathbb{R}_+$ *such that for any* $t \geq A$, $f(t) \leq c e^{-\alpha t}$, *where* $f(t) = \max\{k(t), |\frac{\partial k}{\partial x}|(t), |\frac{\partial^2 k}{\partial x^2}|(t)\}$,

(iv) $k(0) = 0$.

The assumption that the kernel is non-increasing could be dropped, however it makes the proof and the presentation easier. Moreover, in practice the weights of the neighborhood graph which are determined by k are interpreted as similarities. Therefore the usual choice is to take weights which

2. This is the reason why one replaces normal coordinates in the neighborhood of the boundary with normal collar coordinates.

decrease with increasing distance. The fourth condition implies that the graph has no loops.[3] In particular, the kernel is not continuous at the origin. All results hold also without this condition. The advantage of this condition is that some estimators become unbiased. Also let us introduce the helpful notation, $k_h(t) = \frac{1}{h^m} k\left(\frac{t}{h^2}\right)$ where we call h the bandwidth of the kernel. Moreover, we define the following two constants related to the kernel function k,

$$C_1 = \int_{\mathbb{R}^m} k(\|y\|^2)dy < \infty, \quad C_2 = \int_{\mathbb{R}^m} k(\|y\|^2) y_1^2 dy < \infty.$$

We also have some assumptions on the probability measure P.

Assumption 21 *(i) P is absolutely continuous with respect to the natural volume element dV on M,*

(ii) the density p fulfills: $p \in C^3(M)$ and $p(x) > 0, \forall\, x \in M \backslash \partial M$,

(iii) the sample X_i, $i = 1, \ldots, n$ is drawn i.i.d. from P,

Note that condition (i) implies $P(\partial M) = 0$, that is the boundary ∂M is a set of measure zero. We will call the Assumptions 19 on the submanifold, Assumptions 20 on the kernel function, and Assumptions 21 on the probability measure P together the **standard assumptions**.

In the following table we summarize the notation used in the proofs:

$k : \mathbb{R}_+ \to \mathbb{R}_+$	kernel function
$h > 0$	neighborhood/bandwidth parameter
$m \in \mathbb{N}$	dimension of the submanifold M
$k_h(t) = \frac{1}{h^m} k\left(\frac{t}{h^2}\right)$	scaled kernel function
$\lambda \in \mathbb{R}$	reweighting parameter
$d_{h,n}(x) = \frac{1}{n}\sum_{i=1}^n k_h(\|x - X_i\|^2)$	degree function associated with k
$\tilde{k}_{\lambda,h}(x, X_i) = \frac{k_h(\|x-X_i\|^2)}{[d_{h,n}(X_i) d_{h,n}(X_j)]^\lambda}$	reweighted kernel
$\tilde{d}_{\lambda,h,n}(x) = \frac{1}{n}\sum_{i=1}^n \tilde{k}_{\lambda,h}(x, X_i)$	degree function associated with $\tilde{k}_{\lambda,h}$
$(\tilde{A}_{\lambda,h,n} f)(x) = \frac{1}{n}\sum_{i=1}^n \tilde{k}_{\lambda,h}(x, X_i) f(X_i)$	empirical average operator $\tilde{A}_{\lambda,h,n}$
$\Delta^{(\mathrm{rw})}_{\lambda,h,n} f = \frac{1}{h^2}\left(f - \frac{1}{\tilde{d}_{\lambda,h,n}} \tilde{A}_{\lambda,h,n} f\right)$	random walk graph Laplacian
$\Delta^{(\mathrm{u})}_{\lambda,h,n} f = \frac{1}{h^2}\left(\tilde{d}_{\lambda,h,n} f - \tilde{A}_{\lambda,h,n} f\right)$	unnormalized graph Laplacian
$\Delta^{(\mathrm{n})}_{\lambda,h,n} f = \frac{1}{h^2}\left(f - \frac{1}{\sqrt{\tilde{d}_{\lambda,h,n}}} \tilde{A}_{\lambda,h,n}\left(\frac{f}{\sqrt{\tilde{d}_{\lambda,h,n}}}\right)\right)$	normalized graph Laplacian
$C_1 = \int_{\mathbb{R}^m} k(\|y\|^2_{\mathbb{R}^m}) dy, C_2 = \int_{\mathbb{R}^m} k(\|y\|^2_{\mathbb{R}^m}) y_1^2 dy$	characteristic constants of the kernel
$p_h(x) = \mathbb{E}_Z\, k_h(\|x - Z\|^2)$	convolution of p with k_h
$(\tilde{A}_{\lambda,h} f)(x) = \mathbb{E}_Z\, \tilde{k}_{\lambda,h}(x, Z) f(Z)$	average operator $\tilde{A}_{\lambda,h}$
$\Delta^{(\mathrm{rw})}_{\lambda,h}, \Delta^{(\mathrm{u})}_{\lambda,h}, \Delta^{(\mathrm{n})}_{\lambda,h}$	Laplacians associated with $\tilde{A}_{\lambda,h}$
$\Delta_s = \frac{1}{p^s}\,\mathrm{div}(p^s\, \mathrm{grad}) = \frac{1}{p^s} g^{ab} \nabla_a (p^s \nabla_b)$	s-th weighted Laplacian on M

3. An edge from a vertex to itself is called a loop.

5.3 Asymptotics of Euclidean Convolutions on the Submanifold M

The following proposition describes the asymptotic expression of the convolution of a function f on the submanifold M with a kernel function having the Euclidean distance $\|x-y\|$ as its argument with respect to the probability measure P on M. This result is interesting since it shows how the use of the Euclidean distance introduces a curvature effect if one averages a function locally. A similar result has been presented in Coifman and Lafon (2006). We define the density p invariantly with respect to the natural volume element and also explicitly give the second order curvature terms. Our proof is similar to that of Smolyanov, von Weizsäcker, and Wittich (2007) where under stronger conditions a similar result was proven for the Gaussian kernel. The more general setting and the use of general kernel functions make the proof slightly more complicated. In order to emphasize the distinction between extrinsic and intrinsic properties of the manifold we will use the slightly cumbersome notations $x \in M$ (intrinsic) and $i(x) \in \mathbb{R}^d$ (extrinsic).

Proposition 22 *Let M and k satisfy Assumptions 19 and 20. Furthermore, let P have a density p with respect to the natural volume element and $p \in C^3(M)$. Then, for any $x \in M \setminus \partial M$, there exists an $h_0(x) > 0$ such that for all $h < h_0(x)$ and any $f \in C^3(M)$,*

$$\begin{aligned}&\int_M k_h\big(\|i(x)-i(y)\|^2_{\mathbb{R}^d}\big) f(y)p(y)\sqrt{\det g}\,dy \\ =&C_1 p(x)f(x) + \frac{h^2}{2}C_2\Big(p(x)f(x)S(x) + (\Delta_M(pf))(x)\Big) + O(h^3),\end{aligned}$$

where $O(h^3)$ is a function depending on x, $\|f\|_{C^3(M)}$ and $\|p\|_{C^3(M)}$ and

$$S(x) = \frac{1}{2}\Big[-R\big|_x + \frac{1}{2}\,\big\|\textstyle\sum_a \Pi(\partial_a,\partial_a)\big\|^2_{T_{i(x)}\mathbb{R}^d}\Big],$$

where R is the scalar curvature and Π the second fundamental form of M .

The following Lemma is an application of Bernstein's inequality. Together with the previous proposition it will be the main ingredient for proving consistency statements for the graph structure.

Lemma 23 *Suppose the standard assumptions hold and let the kernel k have compact support on $[0,R_k^2]$. Define $b_1 = \|k\|_\infty \|f\|_\infty$, $b_2 = K\|f\|^2_\infty$ where K is a constant depending on $\|p\|_\infty$, $\|k\|_\infty$ and R_k. Let $x \in M\setminus\partial M$ and $V_i := k_h(\|i(x)-i(X_i)\|^2)f(X_i)$. Then for any bounded function f,*

$$\mathrm{P}\Big(\Big|\frac{1}{n}\sum_{i=1}^n V_i - \mathbb{E}V\Big| > \varepsilon\Big) \le 2\exp\Big(-\frac{nh^m\varepsilon^2}{2b_2+2b_1\varepsilon/3}\Big).$$

Let $W_i = k_h(\|i(x)-i(X_i)\|^2)(f(x)-f(X_i))$. Then for $hR_k \le \kappa/2$ and $f \in C^1(M)$,

$$\mathrm{P}\Big(\Big|\frac{1}{n}\sum_{i=1}^n W_i - \mathbb{E}W\Big| > h\,\varepsilon\Big) \le 2\exp\Big(-\frac{nh^m\varepsilon^2}{2b_2+2b_1\varepsilon/3}\Big).$$

Proof Since by assumption $\kappa > 0$, by Lemma 18, for any $x,y \in M$ with $\|i(x)-i(y)\| \le \kappa/2$, we have $d_M(x,y) \le 2\,\|i(x)-i(y)\|$. This implies $\forall a \le \kappa/2$, $B_{\mathbb{R}^d}(x,a)\cap M \subset B_M(x,2a)$.

Let $W_i := k_h(\|i(x)-i(X_i)\|^2)f(X_i)$. We have

$$|W_i| \leq \frac{\|k\|_\infty}{h^m} \sup_{y \in B_{\mathbb{R}^d}(x,hR_k)\cap M} |f(y)| \leq \frac{\|k\|_\infty}{h^m} \|f\|_\infty := \frac{b_1}{h^m}.$$

For the variance of W we have two cases. First let $hR_k < s := \min\{\kappa/2, R_0/2\}$. Then we get,

$$\operatorname{Var} W \leq \mathbb{E}_Z k_h^2(\|i(x)-i(Z)\|^2) f^2(Z) \leq \frac{\|k\|_\infty}{h^m} \|f\|_\infty^2 \, p_h(x) \leq D_2 \frac{\|k\|_\infty}{h^m} \|f\|_\infty^2,$$

where we have used Lemma 46 in the last step. Now consider $hR_k \geq s$, then

$$\operatorname{Var} W \leq \frac{\|k\|_\infty^2}{h^{2m}} \|f\|_\infty^2 \leq \frac{R_k^m \|k\|_\infty^2}{s^m \, h^m} \|f\|_\infty^2.$$

Therefore we define $b_2 = K\|f\|_\infty^2$ with $K = R_k^m \|k\|_\infty^2 \max\{2^m S_2 \|p\|_\infty, s^{-m}\}$. By Bernstein's inequality we finally get

$$\mathrm{P}\Big(\Big|\tfrac{1}{n}\textstyle\sum_{i=1}^n W_i - \mathbb{E}W\Big| > \varepsilon\Big) \quad \leq \quad 2e^{-\frac{nh^m\varepsilon^2}{2b_2+2b_1\varepsilon/3}}.$$

Both constants b_2 and b_1 are independent of x. For the second part note that by Lemma 18 for $hR_k \leq \kappa/2$, we have that $\|x-y\| \leq hR_k$ implies $d_M(x,y) \leq 2\|x-y\| \leq 2hR_k$. In particular, for all $x, y \in M$ with $\|x-y\| \leq hR_k$,

$$|f(x)-f(y)| \leq \sup_{y\in M} \|\nabla f\|_{T_yM} \, d_M(x,y) \leq 2hR_k \sup_{y\in M} \|\nabla f\|_{T_yM}.$$

A similar reasoning as above leads then to the second statement. ■

Note that $\mathbb{E}_Z k_h(\|i(x)-i(Z)\|^2)f(Z) = \int_M k_h(\|i(x)-i(y)\|^2) f(y) p(y) \sqrt{\det g}\, dy$.

5.4 Pointwise Consistency of the Random Walk, Unnormalized and Normalized Graph Laplacian

The proof of the convergence result for the three graph Laplacians is organized as follows. First we introduce the continuous operators $\Delta_{\lambda,h}^{(\mathrm{rw})}$, $\Delta_{\lambda,h}^{(\mathrm{n})}$ and $\Delta_{\lambda,h}^{(\mathrm{u})}$. Then we derive the limit of the continuous operators as $h \to 0$. This part of the proof is concerned with the bias part since roughly $(\Delta_{\lambda,h} f)(x)$ can be seen as the expectation of $\Delta_{\lambda,h,n} f(x)$. Second we show that with high probability all extended graph Laplacians are close to the corresponding continuous operators. This is the variance part. Combining both results we arrive finally at the desired consistency results.

5.4.1 THE BIAS PART - DEVIATION OF $\Delta_{\lambda,h}$ FROM ITS LIMIT

The following continuous approximation of $\Delta_{\lambda,h}^{(\mathrm{rw})}$ was similarly introduced in Lafon (2004) and Coifman and Lafon (2006).

Definition 24 (Kernel-based approximation of the Laplacian) *We introduce the following averaging operator $\tilde{A}_{\lambda,h}$ based on the reweighted kernel $\tilde{k}_{\lambda,h}$,*

$$(\tilde{A}_{\lambda,h} f)(x) = \int_M \tilde{k}_{\lambda,h}(x,y) f(y) p(y) \sqrt{\det g}\, dy,$$

and with $\tilde{d}_{\lambda,h} = (\tilde{A}_{\lambda,h}1)$ *the following continuous operators,*

$$\begin{aligned}
\text{random walk}: &\quad \Delta^{(rw)}_{\lambda,h} f := \frac{1}{h^2}\Big(f - \frac{1}{\tilde{d}_{\lambda,h}}\tilde{A}_{\lambda,h} f\Big) = \frac{1}{h^2}\Big(\frac{\tilde{A}_{\lambda,h} g}{\tilde{d}_{\lambda,h}}\Big)(x),\\
\text{unnormalized}: &\quad \Delta^{(u)}_{\lambda,h} f := \frac{1}{h^2}\Big(\tilde{d}_{\lambda,h} f - \tilde{A}_{\lambda,h} f\Big) = \frac{1}{h^2}(\tilde{A}_{\lambda,h} g)(x),\\
\text{normalized}: &\quad \Delta^{(n)}_{\lambda,h} f := \frac{1}{h^2\sqrt{\tilde{d}_{\lambda,h}}}\Big(d_{\lambda,h}\frac{f}{\sqrt{\tilde{d}_{\lambda,h}}} - \tilde{A}_{\lambda,h}\frac{f}{\sqrt{\tilde{d}_{\lambda,h}}}\Big) = \frac{1}{h^2\sqrt{\tilde{d}_{\lambda,h}(x)}}(\tilde{A}_{\lambda,h} g')(x),
\end{aligned}$$

where we have introduced again $g(y) := f(x) - f(y)$ and $g'(y) := \frac{f(x)}{\sqrt{\tilde{d}_{\lambda,h}(x)}} - \frac{f(y)}{\sqrt{\tilde{d}_{\lambda,h}(y)}}$. The definition of the normalized approximation $\Delta^{(\mathrm{rw})}_{\lambda,h}$ can be justified by the alternative definition of the Laplacian in $\mathbb{R}^d$ sometimes made in physics textbooks:

$$(\Delta f)(x) = \lim_{r\to 0} -\frac{1}{C_d\, r^2}\Big(f(x) - \frac{1}{\mathrm{vol}(B(x,r))}\int_{B(x,r)} f(y)dy\Big),$$

where C_d is a constant depending on the dimension d.

Approximations of the Laplace-Beltrami operator based on averaging with the Gaussian kernel in the case of a uniform probability measure have been studied for compact submanifolds without boundary by Smolyanov, von Weizsäcker, and Wittich (2000, 2007) and Belkin (2003). Their result was then generalized by Lafon (2004) to general densities and to a wider class of isotropic, positive definite kernels for compact submanifolds with boundary. The proof given in Lafon (2004) applies only to compact hypersurfaces[4] in $\mathbb{R}^d$, a proof for the general case of compact submanifolds with boundary using boundary conditions has been presented in Coifman and Lafon (2006). In this section we will prove the pointwise convergence of the continuous approximation for general submanifolds M with boundary of bounded geometry with the additional Assumptions 19. This includes the case where M is not compact. Moreover, no assumptions of positive definiteness of the kernel are made nor any boundary condition on the function f is imposed. Almost any submanifold occurring in practice should be covered in this very general setting.

For pointwise convergence in the interior of the manifold M boundary conditions on f are not necessary. However, for uniform convergence there is no way around them. Then the problem lies not in the proof that the continuous approximation still converges in the right way but in the transfer of the boundary condition to the discrete graph. The main problem is that since we have no information about M apart from the random samples the boundary will be hard to locate. Moreover, since the boundary is a set of measure zero, we will actually almost surely never sample any point from the boundary. The rigorous treatment of the approximation of the boundary respectively the boundary conditions of a function on a randomly sampled graph remains as an open problem.

Especially for dimensionality reduction the case of low-dimensional submanifolds in $\mathbb{R}^d$ is important. Notably, the analysis below also includes the case where due to noise the data is only concentrated around a submanifold.

Theorem 25 *Suppose the standard assumptions hold. Furthermore, let k be a kernel with compact support on* $[0, R_k^2]$. *Let* $\lambda \in \mathbb{R}$, *and* $x \in M\backslash\partial M$. *Then there exists an* $h_1(x) > 0$ *such that for all*

4. A hypersurface is a submanifold of codimension 1.

$h < h_1(x)$ and any $f \in C^3(M)$,

$$(\Delta^{(rw)}_{\lambda,h} f)(x) = -\frac{C_2}{2C_1}\Big((\Delta_M f)(x) + \frac{s}{p(x)}\left\langle \nabla p, \nabla f\right\rangle_{T_xM}\Big) + O(h) = -\frac{C_2}{2C_1}(\Delta_s f)(x) + O(h),$$

where Δ_M is the Laplace-Beltrami operator of M and $s = 2(1-\lambda)$.

Proof For sufficiently small h we have $\overline{B_{\mathbb{R}^d}(x, 2hR_k) \cap M} \cap \partial M = \emptyset$. Moreover, it can be directly seen from the proof of Proposition 22 that the upper bound of the interval $[0, h_0(y)]$ for which the expansion holds depends continuously on $\delta(x)$ and $\varepsilon(y)$, where $\varepsilon(y) = \frac{1}{3}\min\{\pi\rho, \mathrm{inj}(y)\}$. Now $h_0(x)$ is continuous since $\mathrm{inj}(x)$ is continuous on compact subsets, see Klingenberg (1982)[Prop. 2.1.10], and $\delta(x)$ is continuous since the injectivity radius is continuous. Therefore we conclude that since $h_0(y)$ is continuous on $\overline{B(x, 2hR_k) \cap M}$ and $h_0(y) > 0$, $h_1(x) = \inf_{y \in \overline{B_{\mathbb{R}^d}(x, 2hR_k) \cap M}} h_0(y) > 0$. Then for the interval $(0, h_1(x))$ the expansion of $p_h(y)$ holds uniformly over the whole set $B(x, 2hR_k) \cap M$. That is, using the definition of $\tilde{k}$ as well as Proposition 22 and the expansion $\frac{1}{(a+h^2b)^\lambda} = \frac{1}{a^\lambda} - \lambda \frac{h^2 b}{a^{\lambda+1}} + O(h^4)$, we get for $h \in (0, h_1(x))$ that

$$\int_M \tilde{k}_{\lambda,h}\big(\|i(x) - i(y)\|^2\big) f(y) p(y) \sqrt{\det g}\, dy$$
$$= \int\limits_{B_{\mathbb{R}^d}(x, hR_k) \cap M} \frac{k_h\big(\|i(x) - i(y)\|^2\big)}{p_h^\lambda(x)} f(y) \left[\frac{C_1 p(y) - \lambda/2 C_2 h^2 (p(y)S + \Delta p)}{C_1^{\lambda+1} p(y)^\lambda} + O(h^3)\right] \sqrt{\det g}\, dy,$$

where the $O(h^3)$-term is continuous on $B_{\mathbb{R}^d}(x, hR_k)$ and we have introduced the abbreviation $S = \frac{1}{2}[-R + \frac{1}{2}\left\|\sum_a \Pi(\partial_a, \partial_a)\right\|^2_{T_{i(x)}\mathbb{R}^d}]$. Using $f(y) = 1$ we get,

$$\tilde{d}_{\lambda,h}(x) = \int\limits_{B_{\mathbb{R}^d}(x, hR_k) \cap M} \frac{k_h\big(\|i(x) - i(y)\|^2\big)}{p_h^\lambda(x)} \left[\frac{C_1 p(y) - \lambda/2 C_2 h^2 (p(y)S + \Delta p)}{C_1^{\lambda+1} p(y)^\lambda} + O(h^3)\right] \sqrt{\det g}\, dy,$$

as an estimate for $\tilde{d}_{\lambda,h}(x)$. Now using Proposition 22 again, we arrive at:

$$\Delta^{(\mathrm{rw})}_{\lambda,h} f = \frac{1}{h^2} \frac{\tilde{d}_{\lambda,h} f - \tilde{A}_{\lambda,h} f}{\tilde{d}_{\lambda,h}} = -\frac{C_2}{2C_1}\Big(\Delta_M f + \frac{2(1-\lambda)}{p}\left\langle \nabla p, \nabla f\right\rangle\Big) + O(h),$$

where all $O(h)$-terms are finite on $B_{\mathbb{R}^d}(x, hR_k) \cap M$ since p is strictly positive. ■

Note that the limit of $\Delta^{(\mathrm{rw})}_{\lambda,h}$ has the opposite sign of Δ_s. This is due to the fact that the Laplace-Beltrami operator on manifolds is usually defined as a negative definite operator (in analogy to the Laplace operator in $\mathbb{R}^d$), whereas the graph Laplacian is positive definite. But this varies through the literature, thus the reader should be aware of the sign convention.

Remark 26 *The assumption of compact support of the kernel k is only necessary in the case of non-compact manifolds M. For compact manifolds a kernel with non-compact support, such as a Gaussian kernel, would work, too. The reason for compact support of the kernel comes from the fact that for non-compact manifolds there exists no lower bound on a strictly positive density. This in turn implies that one cannot upper bound the convolution with the reweighted kernel if one does not impose additional assumptions on the density. In practice the solution of graph-based methods for large-scale problems is usually only possible for sparse neighborhood graphs. Therefore the compactness assumption of the kernel is quite realistic and does not exclude relevant cases.*

With the relations,

$$(\Delta^{(\mathrm{u})}_{\lambda,h,n}f)(x) = \tilde{d}_{\lambda,h,n}(x)(\Delta^{(\mathrm{rw})}_{\lambda,h,n}f)(x),$$
$$(\Delta^{(\mathrm{n})}_{\lambda,h,n}f)(x) = 1/\sqrt{\tilde{d}_{\lambda,h,n}(x)}\,\Big(\Delta^{(\mathrm{u})}_{\lambda,h,n}\Big(f/\sqrt{\tilde{d}_{\lambda,h,n}}\Big)\Big)(x),$$

one can easily adapt the last lines of the previous proof to derive the following corollary.

Corollary 27 *Under the assumptions of Theorem 25. Let $\lambda \in \mathbb{R}$ and $x \in M\backslash\partial M$. Then there exists an $h_1(x) > 0$ such that for all $h < h_1(x)$ and any $f \in C^3(M)$,*

$$(\Delta^{(u)}_{\lambda,h}f)(x) = -\frac{C_2}{2C_1^{2\lambda}}\,p(x)^{1-2\lambda}(\Delta_s f)(x) + O(h), \quad \text{where} \quad s = 2(1-\lambda),$$
$$(\Delta^{(n)}_{\lambda,h}f)(x) = -\frac{C_2}{2C_1}\,p(x)^{\frac{1}{2}-\lambda}\Delta_s\left(\frac{f}{p^{\frac{1}{2}-\lambda}}\right)(x) + O(h).$$

5.4.2 THE VARIANCE PART - DEVIATION OF $\Delta_{\lambda,h,n}$ FROM $\Delta_{\lambda,h}$

Before we state the results for the general case with data-dependent weights we now treat the case $\lambda = 0$, that is we have non-data-dependent weights. There the proof is considerably simpler and much easier to follow. Moreover, as opposed to the general case here we get convergence in probability under slightly weaker conditions than almost sure convergence. Since this does not hold for the normalized graph Laplacian in that case we will only provide the general proof.

Theorem 28 (Weak and strong pointwise consistency for $\lambda = 0$) *Suppose the standard assumptions hold. Furthermore, let k be a kernel with compact support on $[0, R_k^2]$. Let $x \in M\backslash\partial M$ and $f \in C^3(M)$. Then if $h \to 0$ and $nh^{m+2} \to \infty$,*

$$\lim_{n\to\infty}(\Delta^{(rw)}_{0,h,n}f)(x) = -\frac{C_2}{2C_1}(\Delta_2 f)(x) \qquad \textit{in probability},$$
$$\lim_{n\to\infty}(\Delta^{(u)}_{0,h,n}f)(x) = -\frac{C_2}{2}\,p(x)(\Delta_2 f)(x) \qquad \textit{in probability}.$$

If even $nh^{m+2}/\log n \to \infty$, then almost sure convergence holds.

Proof We give the proof for $\Delta^{(\mathrm{rw})}_{0,h,n}$. The proof for $\Delta^{(\mathrm{u})}_{0,h,n}$ can be directly derived with the second statement of Lemma 23 for the variance term together with Corollary 27 for the bias term. Similar to the proof for the Nadaraya-Watson regression estimate of Greblicki et al. (1984), we rewrite the estimator $\Delta^{(\mathrm{rw})}_{0,h,n}f$ in the following form

$$(\Delta^{(\mathrm{rw})}_{0,h,n}f)(x) = \frac{1}{h^2}\left[\frac{(C_{0,h}f)(x) + B_{1n}}{1 + B_{2n}}\right],$$

where

$$(C_{0,h}f)(x) = \frac{\mathbb{E}_Z\, k_h(\|i(x)-i(Z)\|^2)g(Z)}{\mathbb{E}_Z\, k_h(\|i(x)-i(Z)\|^2)},$$
$$B_{1n} = \frac{\frac{1}{n}\sum_{j=1}^n k_h(\|i(x)-i(X_j)\|^2)g(X_j) - \mathbb{E}_Z\, k_h(\|i(x)-i(Z)\|^2)g(Z)}{\mathbb{E}_Z\, k_h(\|i(x)-i(Z)\|^2)},$$
$$B_{2n} = \frac{\frac{1}{n}\sum_{j=1}^n k_h(\|i(x)-i(X_j)\|^2) - \mathbb{E}_Z\, k_h(\|i(x)-i(Z)\|^2)}{\mathbb{E}_Z\, k_h(\|i(x)-i(Z)\|^2)},$$

with $g(y) := f(x) - f(y)$. In Theorem 25 we have shown that for $x \in M \backslash \partial M$,

$$\lim_{h \to 0} (\Delta^{(\mathrm{rw})}_{0,h} f)(x) = \lim_{h \to 0} \frac{1}{h^2} (C_{0,h}\, g)(x) = -\frac{C_2}{2C_1} (\Delta_2 f)(x).$$

Using the lower bound of $p_h(x) = \mathbb{E}_Z\, k_h(\|i(x) - i(Z)\|^2)$ derived in Lemma 46 we can for $hR_k \leq \kappa/2$ directly apply Lemma 23. Thus there exist constants d_1 and d_2 such that

$$\mathrm{P}(\,|B_{1n}| \geq h^2 t\,) \leq 2 \exp\left(-\frac{nh^{m+2}t^2}{2\,\|k\|_\infty (d_2 + t\,h\,d_1/3)}\right).$$

The same analysis can be done for B_{2n}. This shows convergence in probability. Complete convergence (which implies almost sure convergence) can be shown by proving for all $t > 0$ the convergence of the series $\sum_{n=0}^{\infty} \mathrm{P}\left(|B_{1n}| \geq h^2 t\right) < \infty$. A sufficient condition for that is $nh^{m+2}/\log n \to \infty$ as $n \to \infty$. ■

The weak pointwise consistency of the unnormalized graph Laplacian for compact submanifolds with the uniform probability measure using the Gaussian kernel for the weights and $\lambda = 0$ was proven by Belkin and Niyogi (2005). A more general result appeared independently in Hein et al. (2005). We prove here the limits of all three graph Laplacians for general submanifolds with boundary of bounded geometry, general probability measures P, and general kernel functions k as stated in our standard assumptions.

The rest of this section is devoted to the general case $\lambda \neq 0$. We show that with high probability the extended graph Laplacians $\Delta_{\lambda,h,n}$ are pointwise close to the continuous operators $\Delta_{\lambda,h}$ when applied to a function $f \in C^3(M)$. The following proposition is helpful.

Proposition 29 *Suppose the standard assumptions hold. Furthermore, let k be a kernel with compact support on $[0, R_k^2]$. Fix $\lambda \in \mathbb{R}$ and let $x \in M \backslash \partial M$, $f \in C^3(M)$ and define $g(y) := f(x) - f(y)$. Then there exists a constant C such that for any $\frac{2\|k\|_\infty}{nh^m} < \varepsilon < 1/C$, $0 < h < \frac{\kappa}{2R_k}$, the following events hold with probability at least $1 - C\,n\,e^{\frac{-nh^m \varepsilon^2}{C}}$,*

$$|(\tilde{A}_{\lambda,h,n} g)(x) - (\tilde{A}_{\lambda,h} g)(x)| \leq \varepsilon\, h, \qquad |\tilde{d}_{\lambda,h,n}(x) - \tilde{d}_{\lambda,h}(x)| \leq \varepsilon.$$

Proof The idea of this proof is to show that several empirical quantities which can be expressed as a sum of i.i.d. random variables are close to their expectation. Then one can deduce that also $(\tilde{A}_{\lambda,h,n} g)(x)$ will be close to $(\tilde{A}_{\lambda,h} g)(x)$. The proof for $\tilde{d}_{\lambda,h,n}$ can then be easily adapted from the following. We consider here only $\lambda \geq 0$, the proof for $\lambda < 0$ is even simpler. Consider the event $\mathcal{E}$ for which one has

$$\begin{cases} \text{for any } j \in \{1, \dots, n\},\ \left|d_{h,n}(X_j) - p_h(X_j)\right| \leq \varepsilon, \\ \left|d_{h,n}(x) - p_h(x)\right| \leq \varepsilon, \\ \left|\frac{1}{n} \sum\limits_{j=1}^{n} \frac{k_h(\|i(x) - i(X_j)\|^2)|g(X_j)|}{[p_h(x)\,p_h(X_j)]^\lambda} - \int\limits_M k_h(\|i(x) - i(y)\|^2)|g(y)| \frac{p(y)}{[p_h(x)\,p_h(y)]^\lambda} \sqrt{\det g}\, dy\right| \leq h\,\varepsilon. \end{cases}$$

We will now prove that for sufficiently large C the event $\mathcal{E}$ holds with probability at least $1 - Cne^{-\frac{nh^m\varepsilon^2}{C}}$. For the second assertion defining $\mathcal{E}$, we use Lemma 23

$$\mathrm{P}(\,|d_{h,n}(x) - p_h(x)| > \varepsilon\,) \leq 2 \exp\left(-\frac{nh^m \varepsilon^2}{2b_2 + 2b_1 \varepsilon/3}\right),$$

where b_1 and b_2 are constants depending on the kernel k and p. For the first term in the event $\mathcal{E}$ remember that $k(0)=0$. We get for $\frac{\|k\|_\infty}{nh^m} < \varepsilon/2$ and $1 \le j \le n$,

$$P\Big(\big|\tfrac{1}{n}\sum_{i=1}^n k_h(\|i(X_j)-i(X_i)\|^2) - p_h(x)\big| > \varepsilon \,\Big|\, X_j\Big) \le 2\exp\Big(-\tfrac{(n-1)h^m\varepsilon^2}{8b_2+4b_1\varepsilon/3}\Big).$$

This follows by

$$\Big|\frac{1}{n}\sum_{i=1}^n k_h(\|i(X_j)-i(X_i)\|^2) - p_h(X_j)\Big| \le \Big|\frac{1}{n(n-1)}\sum_{i=1}^n k_h(\|i(X_j)-i(X_i)\|^2)\Big| + \Big|\frac{1}{n-1}\sum_{i\neq j} k_h(\|i(X_j)-i(X_i)\|^2) - p_h(X_j)\Big|,$$

where the first term is upper bounded by $\frac{\|k\|_\infty}{nh^m}$. First integrating wrt to the law of X_j (the right hand side of the bound is independent of X_j) and then using a union bound, we get

$$P\Big(\text{for any } j\in\{1,\dots,n\},\ \big|d_{h,n}(X_j)-p_h(X_j)\big| \le \varepsilon\Big) > 1-2n\exp\Big(-\tfrac{(n-1)h^m\varepsilon^2}{8b_2+4b_1\varepsilon/3}\Big).$$

Noting that $\frac{1}{p_h(x)p_h(y)}$ is upper bounded by Lemma 46 we get by Lemma 23 for $hR_k \le \kappa/2$ a Bernstein type bound for the probability of the third event in $\mathcal{E}$. Finally, combining all these results, we obtain that there exists a constant C such that for $h \le \frac{\kappa}{2R_k}$ and $\frac{2\|k\|_\infty}{nh^m} \le \varepsilon \le 1$, the event[5] $\mathcal{E}$ holds with probability at least $1-Cne^{-\frac{nh^m\varepsilon^2}{C}}$. Let us define

$$\begin{cases} \mathcal{B} := \int_M k_h(\|i(x)-i(y)\|^2)(f(x)-f(y))[p_h(x)\,p_h(y)]^{-\lambda}p(y)\sqrt{\det g}\,dy, \\ \hat{\mathcal{B}} := \frac{1}{n}\sum_{j=1}^n k_h(\|i(x)-i(X_j)\|^2)(f(x)-f(X_j))\big[d_{h,n}(x)\,d_{h,n}(X_j)\big]^{-\lambda}, \end{cases}$$

then $(\tilde{A}_{\lambda,h,n}g)(x)=\hat{\mathcal{B}}$ and $(\tilde{A}_{\lambda,h}g)(x)=\mathcal{B}$. Let us now work only on the event $\mathcal{E}$. By Lemma 46 for any $y \in B_{\mathbb{R}^d}(x,hR_k)\cap M$ there exist constants D_1,D_2 such that $0<D_1\le p_h(y)\le D_2$. Using the first order Taylor formula of $[x\mapsto x^{-\lambda}]$, we obtain that for any $\lambda\ge 0$ and $a,b>\beta$, $\big|a^{-\lambda}-b^{-\lambda}\big| \le \lambda\beta^{-\lambda-1}|a-b|$. So we can write for $\varepsilon < D_1/2$,

$$\begin{aligned}\Big|\frac{1}{\big(d_{h,n}(x)\,d_{h,n}(X_j)\big)^\lambda} - \frac{1}{\big(p_h(x)\,p_h(X_j)\big)^\lambda}\Big| &\le \lambda(D_1-\varepsilon)^{-2\lambda-2}|d_{h,n}(x)d_{h,n}(X_j)-p_h(x)p_h(X_j)| \\ &\le 2\lambda(D_1-\varepsilon)^{-2\lambda-2}(D_2+\varepsilon)\varepsilon := C\varepsilon.\end{aligned}$$

Noting that for $hR_k\le\kappa/2$ by Lemma 18, $d_M(x,y)\le 2hR_k$, $\forall y\in B_{\mathbb{R}^d}(x,hR_k)\cap M$,

$$\begin{aligned}\big|\hat{\mathcal{B}}-\mathcal{B}\big| &\le \Big|\tfrac{1}{n}\sum_{j=1}^n k_h(\|i(x)-i(X_j)\|^2)|f(x)-f(X_j)|C\varepsilon \\ &\quad + \Big|\tfrac{1}{n}\sum_{j=1}^n k_h(\|i(x)-i(X_j)\|^2)(f(x)-f(X_j))[p_h(x)\,p_h(X_j)]^{-\lambda} - \mathcal{B}\Big| \\ &\le 2C\,\|k\|_\infty R_k \sup_{y\in M}\|\nabla f\|_{T_yM}\,h\varepsilon + h\varepsilon.\end{aligned}$$

We have proven that there exists a constant $C>1$ such that for any $0<h<\frac{\kappa}{2R_k}$ and $\frac{2\|k\|_\infty}{nh^m}<\varepsilon<1/C$,

$$\big|(\tilde{A}_{\lambda,h,n}g)(x)-(\tilde{A}_{\lambda,h}g)(x)\big| \le C'''h\varepsilon,$$

5. The upper bound on ε is here not necessary but allows to write the bound more compactly.

with probability at least $1 - Cne^{-\frac{nh^m\varepsilon^2}{C}}$. ■

This leads us to our first main result for the random walk and the unnormalized graph Laplacian.

Theorem 30 (Pointwise consistency of $\Delta^{(\mathbf{rw})}_{\lambda,h,n}$ and $\Delta^{(\mathbf{u})}_{\lambda,h,n}$) *Suppose the standard assumptions hold. Furthermore, let k be a kernel with compact support on $[0, R_k^2]$. Let $x \in M \backslash \partial M$, $\lambda \in \mathbb{R}$. Then for any $f \in C^3(M)$ there exists a constant C such that for any $\frac{2\|k\|_\infty}{nh^{m+1}} < \varepsilon < 1/C$, $0 < h < h_{\max}$ with probability at least $1 - C\,n\,e^{\frac{-nh^{m+2}\varepsilon^2}{C}}$,*

$$|(\Delta^{(rw)}_{\lambda,h,n} f)(x) - (\Delta^{(rw)}_{\lambda,h} f)(x)| \leq \varepsilon,$$
$$|(\Delta^{(u)}_{\lambda,h,n} f)(x) - (\Delta^{(u)}_{\lambda,h} f)(x)| \leq \varepsilon.$$

Define $s = 2(1-\lambda)$. Then if $h \to 0$ and $nh^{m+2}/\log n \to \infty$,

$$\lim_{n\to\infty} (\Delta^{(rw)}_{\lambda,h,n} f)(x) = -\frac{C_2}{2C_1}(\Delta_s f)(x) \qquad \textit{almost surely},$$
$$\lim_{n\to\infty} (\Delta^{(u)}_{\lambda,h,n} f)(x) = -\frac{C_2}{2C_1^{2\lambda}}\, p(x)^{1-2\lambda}\,(\Delta_s f)(x) \qquad \textit{almost surely}.$$

In particular, under the above conditions,

$$\left|(\Delta^{(rw)}_{\lambda,h,n} f)(x) - \left[-\frac{C_2}{2C_1}(\Delta_s f)(x)\right]\right| = O(h) + O\left(\sqrt{\frac{\log n}{nh^{m+2}}}\right) \qquad a.s.,$$
$$\left|(\Delta^{(u)}_{\lambda,h,n} f)(x) - \left[-\frac{C_2}{2C_1^{2\lambda}}\, p(x)^{1-2\lambda}(\Delta_s f)(x)\right]\right| = O(h) + O\left(\sqrt{\frac{\log n}{nh^{m+2}}}\right) \qquad a.s..$$

The optimal rate for $h(n)$ is $h = O((\log n/n)^{\frac{1}{m+4}})$.

Proof In Equation (3) it was shown that

$$(\Delta^{(\mathrm{rw})}_{\lambda,h,n} f)(x) = \frac{1}{h^2}\left(\frac{\tilde{A}_{\lambda,h,n} g}{\tilde{d}_{\lambda,h,n}}\right)(x), \quad (\Delta^{(\mathrm{u})}_{\lambda,h,n} f)(x) = \frac{1}{h^2}(\tilde{A}_{\lambda,h,n} g)(x),$$

where $g(y) := f(x) - f(y)$. Since f is Lipschitz we can directly apply Proposition 29 so that for the unnormalized Laplacian we get with probability $1 - C\,n\,e^{\frac{-nh^{m+2}\varepsilon^2}{C}}$,

$$|(\Delta^{(\mathrm{u})}_{\lambda,h,n} f)(x) - (\Delta^{(\mathrm{u})}_{\lambda,h} f)(x)| \leq \varepsilon.$$

For the random walk Laplacian $\Delta^{(\mathrm{rw})}_{\lambda,h,n}$ we work on the event where $|\tilde{d}_{\lambda,h,n} - \tilde{d}_{\lambda,h}| \leq h\varepsilon$, where $\varepsilon \leq \frac{1}{2h}\tilde{d}_{\lambda,h}$. This holds by Proposition 29 with probability $1 - C\,n\,e^{\frac{-nh^{m+2}\varepsilon^2}{C}}$. Moreover, note that by Lemmas 18 and 14 for $hR_k \leq \min\{\kappa/2, R_0\}$, we have

$$\left|\tilde{A}_{\lambda,h} g\right| \leq \frac{2^m R_k^m S_2}{D_1^{2\lambda}}\, \|p\|_\infty\, \|k\|_\infty\, 2L(f)\, hR_k.$$

Using Proposition 29 for $\tilde{A}_{\lambda,h,n}g$ and the bounds of $p_h(x)$ from Lemma 46,

$$\begin{aligned}\big|(\Delta^{(\mathrm{rw})}_{\lambda,h,n}f)(x)-(\Delta^{(\mathrm{rw})}_{\lambda,h}f)(x)\big| &= \frac{1}{h^2}\left|\tfrac{(\tilde{A}_{\lambda,h,n}g)(x)}{\tilde{d}_{\lambda,h,n}(x)}-\tfrac{(\tilde{A}_{\lambda,h}g)(x)}{\tilde{d}_{\lambda,h}(x)}\right|\\ &\leq \frac{1}{h^2}\left(\frac{|(\tilde{A}_{\lambda,h,n}g)(x)-(\tilde{A}_{\lambda,h}g)(x)|}{\tilde{d}_{\lambda,h,n}(x)}+(\tilde{A}_{\lambda,h}g)(x)\,\frac{|\tilde{d}_{\lambda,h,n}(x)-\tilde{d}_{\lambda,h}(x)|}{\tilde{d}_{\lambda,h,n}(x)\tilde{d}_{\lambda,h}(x)}\right)\\ &\leq \frac{2D_2^{2\lambda}}{D_1}\,\varepsilon+\frac{2^m R_k^m S_2}{D_1^{2\lambda}}\,\|p\|_\infty\,\|k\|_\infty\,2L(f)\,R_k\,\varepsilon := C\,\varepsilon,\end{aligned}$$

with probability $1-C\,n\,e^{\frac{-nh^{m+2}\varepsilon^2}{C}}$. By Theorem 25 and 27 we have for $s=2(1-\lambda)$,

$$\left|(\Delta^{(\mathrm{rw})}_{\lambda,h}f)(x)-\left[-\frac{C_2}{2C_1}(\Delta_s f)(x)\right]\right|\leq C\,h,$$

$$\left|(\Delta^{(\mathrm{u})}_{\lambda,h}f)(x)-\left[-\frac{C_2}{2C_1^{2\lambda}}\,p(x)^{1-2\lambda}(\Delta_s f)(x)\right]\right|\leq C\,h.$$

Combining both results together with the Borel-Cantelli-Lemma yields almost sure convergence. The optimal rate for $h(n)$ follows by equating both order terms. ■

Using the relationship between the unnormalized and the normalized Laplacian the pointwise consistency can be easily derived. However, the conditions for convergence are slightly stronger since the Laplacian is applied to the function $f/\sqrt{\tilde{d}_{\lambda,h,n}}$.

Theorem 31 (Pointwise consistency of $\Delta^{(\mathrm{n})}_{\lambda,h,n}$) *Suppose that the standard assumptions hold. Furthermore, let k be a kernel with compact support on $[0,R_k^2]$. Let $x\in M\backslash\partial M$, $\lambda\in\mathbb{R}$. Then for any $f\in C^3(M)$ there exists a constant C such that for any $\frac{2\|k\|_\infty}{nh^{m+2}}<\varepsilon<1/C$, $0<h<h_{\max}$ with probability at least $1-C\,n^2\,e^{\frac{-nh^{m+4}\varepsilon^2}{C}}$,*

$$\left|(\Delta^{(n)}_{\lambda,h,n}f)(x)-(\Delta^{(n)}_{\lambda,h}f)(x)\right|\leq\varepsilon.$$

Define $s=2(1-\lambda)$. Then if $h\to 0$ and $nh^{m+4}/\log n\to\infty$,

$$\lim_{n\to\infty}(\Delta^{(n)}_{\lambda,h,n}f)(x)=-p(x)^{\frac{1}{2}-\lambda}\frac{C_2}{2C_1}\Delta_s\left(\frac{f}{p^{\frac{1}{2}-\lambda}}\right)(x)\quad \textit{almost surely.}$$

Proof We reduce the case of $\Delta^{(\mathrm{n})}_{\lambda,h,n}$ to the case of $\Delta^{(\mathrm{u})}_{\lambda,h,n}$. We work on the event where

$$|\tilde{d}_{\lambda,h,n}(x)-\tilde{d}_{\lambda,h}(x)|\leq h^2\,\varepsilon,\qquad |\tilde{d}_{\lambda,h,n}(X_i)-\tilde{d}_{\lambda,h}(X_i)|\leq h^2\,\varepsilon,\quad \forall\, i=1,\ldots,n.$$

From Proposition 29 we know that this holds with probability at least $1-C\,n^2\,e^{\frac{-nh^{m+4}\varepsilon^2}{C}}$. Working on this event we get by a similar argumentation as in the proof of Theorem 30 that there exists a constant C' such that

$$\begin{aligned}\left|(\Delta^{(\mathrm{n})}_{\lambda,h,n}f)(x)-\tfrac{1}{\tilde{d}_{\lambda,h}(x)}\left(\Delta^{(\mathrm{u})}_{\lambda,h,n}\tfrac{f}{\sqrt{\tilde{d}_{\lambda,h}}}\right)(x)\right| &= \frac{1}{h^2}\Big|\tilde{d}_{\lambda,h,n}(x)f(x)\left[\tfrac{1}{\tilde{d}_{\lambda,h}(x)}-\tfrac{1}{\tilde{d}_{\lambda,h,n}(x)}\right]\\ &+\sum_{i=1}^n\tilde{k}_{\lambda,h}(x,X_i)f(X_i)\left[\tfrac{1}{\sqrt{\tilde{d}_{\lambda,h}(x)\tilde{d}_{\lambda,h}(X_i)}}-\tfrac{1}{\sqrt{\tilde{d}_{\lambda,h,n}(x)\tilde{d}_{\lambda,h,n}(X_i)}}\right]\Big|\leq C'\varepsilon.\end{aligned}$$

Noting that $\frac{f}{\tilde{d}_{\lambda,h}}$ is Lipschitz since f and $\tilde{d}_{\lambda,h}$ are Lipschitz and upper and lower bounded, on $M \cap B_{\mathbb{R}^d}(x, hR_k)$ one can apply Theorem 30 to derive the first statement. The second statement follows by Corollary 27. ■

Acknowledgments

We would like to thank Olaf Wittich and Bernhard Schölkopf for helpful comments.

Appendix A. Basic Concepts of Differential Geometry

In this section we introduce the necessary basics of differential geometry, in particular normal coordinates and submanifolds in $\mathbb{R}^d$, used in this paper. Note that the definition of the Riemann curvature tensor varies across textbooks which can result in sign-errors. Throughout the paper we use the convention of Lee (1997).

A.1 Basics

Definition 32 *A d-dimensional **manifold** X **with boundary** is a topological (Hausdorff) space such that every point has a neighborhood homeomorphic to an open subset of $\mathbb{H}^d = \{(x^1, \dots, x^d) \in \mathbb{R}^d \big| x_1 \geq 0\}$. A **chart** (or local coordinate system) (U, ϕ) of a manifold X is an open set $U \subset X$ together with a homeomorphism $\phi : U \to V$ of U onto an open subset $V \subset \mathbb{H}^d$. The coordinates $(x^1, \dots, x^d)$ of $\phi(x)$ are called the coordinates of x in the chart (U, ϕ). A C^r**-atlas** $\mathcal{A}$ is a collection of charts,*

$$\mathcal{A} \triangleq \cup\{(U_\alpha, \phi_\alpha), \alpha \in I\},$$

*where I is an index set, such that $X = \cup_{\alpha \in I} U_\alpha$ and for any $\alpha, \beta \in I$ the corresponding **transition map**,*

$$\phi_\beta \circ \phi_\alpha^{-1}\big|_{\phi_\alpha(U_\alpha \cap U_\beta)} : \phi(U_\alpha \cap U_\beta) \to \mathbb{H}^d,$$

*is r-times continuously differentiable. A **smooth manifold with boundary** is a manifold with boundary with a C^∞-atlas.*

For more technical details behind the definition of a manifold with boundary we refer to Lee (2003). Note that the boundary ∂M of M is a $(d-1)$-dimensional manifold without boundary. In textbooks one often only finds the definition of a manifold without boundary which can be easily recovered from the above definition by replacing $\mathbb{H}^d$ with $\mathbb{R}^d$. The interior $M \backslash \partial M$ of the manifold M is a manifold without boundary.

Definition 33 *A subset M of a d-dimensional manifold X is a m-dimensional **submanifold** M **with boundary** if every point $x \in M$ is in the domain of a chart (U, ϕ) of X such that*

$$\phi : U \cap M \to \mathbb{H}^m \times a, \quad \phi(x) = (x^1, \dots, x^m, a^1, \dots, a^{d-m}),$$

*where a is a fixed element in $\mathbb{R}^{d-m}$. X is called the **ambient space** of M.*

This definition excludes irregular cases like intersecting submanifolds or self-approaching submanifolds. In the following it is more appropriate to take the following point of view. Let M be an m-dimensional manifold. The smooth mapping $i : M \to X$ is said to be an immersion if i is differentiable and the differential of i has rank m everywhere. An injective immersion is called embedding if it is an homeomorphism onto its image. In this case $i(M)$ is a submanifold of X. If M is compact and i is an injective immersion, then i is an embedding. This is not the case if M is not compact since $i(M)$ can be self-approaching.

Definition 34 *A* ***Riemannian manifold*** (M,g) *is a smooth manifold M together with a tensor*[6] *of type $(0,2)$, called the metric tensor g, at each $p \in M$, such that g defines an inner product on the tangent space T_pM which varies smoothly over M. The volume form induced by g is given in local coordinates as $dV = \sqrt{\det g}\, dx^1 \wedge \ldots \wedge dx^m$. dV is uniquely determined by $dV(e_1,\ldots,e_m) = 1$ for any oriented orthonormal basis $e_1,\ldots,e_m$ in T_xM.*

The metric tensor induces for every $p \in M$ an isometric isomorphism between the tangent space T_pM and its dual T_p^*M. A submanifold M of a Riemannian manifold (X,g) has a natural Riemannian metric h induced from X in the following way. Let $i : M \to X$ be an embedding so that M is a submanifold of X. Then one can induce a metric h on M using the mapping i, namely $h = i^*g$, where $i^* : T^*_{i(x)}X \to T^*_xM$ is the pull-back[7] of the differentiable mapping i. In this case i trivially is an isometric embedding of (M,h) into (X,g). In the paper we always use on the submanifold M the metric induced from $\mathbb{R}^d$.

Definition 35 *The* ***Laplace-Beltrami operator*** Δ_M *of a Riemannian manifold is defined as* $\Delta_M = \mathrm{div}(\mathrm{grad})$. *For a twice differentiable function $f : M \to \mathbb{R}$ it is explicitly given as*

$$\Delta_M f = \frac{1}{\sqrt{\det g}} \frac{\partial}{\partial x^j}\Big(\sqrt{\det g}\, g^{ij} \frac{\partial f}{\partial x^i}\Big),$$

where g^{ij} are the components of the inverse of the metric tensor $g = g_{ij}\, dx^i \otimes dx^j$.

A.2 Normal Coordinates

Since in the proofs we use normal coordinates, we give here a short introduction. Intuitively, normal coordinates around a point p of an m-dimensional Riemannian manifold M are coordinates chosen such that M looks around p like $\mathbb{R}^m$ in the best possible way. This is achieved by adapting the coordinate lines to geodesics through the point p. The reference for the following material is the book of Jost (2002). We denote by c_v the unique geodesic starting at $c(0) = x$ with tangent vector $\dot{c}(0) = v$ (c_v depends smoothly on p and v).

Definition 36 *Let M be a Riemannian manifold, $p \in M$, and $V_p = \{v \in T_pM,\ c_v \text{ defined on } [0,1]\}$, then, $\exp_p : V_p \to M$, $v \mapsto c_v(1)$, is called the* ***exponential map*** *of M at p.*

It can be shown that $\exp_p$ maps a neighborhood of $0 \in T_pM$ diffeomorphically onto a neighborhood U of $p \in M$. This justifies the definition of normal coordinates.

6. A tensor T of type (m,n) is a multilinear form $T_pM \times \ldots T_pM \times T_p^*M \times \ldots \times T_p^*M \to \mathbb{R}$ (n-times T_pM, m-times T_p^*M).
7. T_x^*M is the dual of the tangent space T_xM. Every differentiable mapping $i : M \to X$ induces a pull-back $i^* : T^*_{i(x)}X \to T_x^*M$. Let $u \in T_xM$, $w \in T^*_{i(x)}X$ and denote by i' the differential of i. Then i^* is defined by $(i^*w)(u) = w(i'u)$.

Definition 37 *Let U be a neighborhood of p in M such that* $\exp_p$ *is a diffeomorphism. The local coordinates defined by the chart* $(U, \exp_p^{-1})$ *are called* ***normal coordinates*** *at p.*

Note that in $T_pM \simeq \mathbb{R}^m \supset \exp_p^{-1}(U)$ we use always an orthonormal basis. The injectivity radius describes the largest ball around p such that normal coordinates can be introduced.

Definition 38 *Let M be a Riemannian manifold. The* ***injectivity radius*** *of $p \in M$ is*

$$\mathrm{inj}(p) = \sup\{\rho > 0,\ \exp_p \text{ is defined on } \overline{B_{\mathbb{R}^m}(0,\rho)} \text{ and injective}\}.$$

It can be shown that $\mathrm{inj}(p) > 0, \forall p \in M \backslash \partial M$. Moreover, for compact manifolds without boundary there exists a lower bound $\mathrm{inj}_{\min} > 0$ such that $\mathrm{inj}(p) \geq \mathrm{inj}_{\min}, \forall p \in M$. However, for manifolds with boundary one has $\mathrm{inj}(p_n) \to 0$ for any sequence of points p_n with limit on the boundary. The motivation for introducing normal coordinates is that the geometry is particularly simple in these coordinates. The following theorem makes this more precise.

Theorem 39 *In normal coordinates around p one has for the Riemannian metric g and the Laplace-Beltrami operator Δ_M applied to a function f at $p = \exp_p^{-1}(0)$,*

$$g_{ij}(0) = \delta_{ij}, \quad \frac{\partial}{\partial x^k} g_{ij}(0) = 0, \quad (\Delta_M f)(0) = \sum_{i=1}^{m} \frac{\partial^2 f}{\partial (x^i)^2}(0).$$

The second derivatives of the metric tensor cannot be made to vanish in general. There curvature effects come into play which cannot be deleted by a coordinate transformation. To summarize, normal coordinates with center p achieve that, up to first order, the geometry of M at point p looks like that of $\mathbb{R}^m$.

A.3 The Second Fundamental Form

In this section we assume that M is an isometrically embedded submanifold of a manifold X. At each point $p \in M$ one can decompose the tangent space T_pX into a subspace T_pM, which is the tangent space to M, and the orthogonal normal space N_pM. In the same way one can split the covariant derivative of X at p, $\tilde{\nabla}_U V$ into a component tangent $(\tilde{\nabla}_U V)^\top$ and normal $(\tilde{\nabla}_U V)^\perp$ to M.

Definition 40 *The* ***second fundamental form*** Π *of an isometrically embedded submanifold M of X is defined as*

$$\Pi : T_pM \otimes T_pM \to N_pM, \quad \Pi(U,V) = (\tilde{\nabla}_U V)^\perp.$$

The following theorem, see Lee (1997), then shows that the covariant derivative of M at p is nothing else than the projection of the covariant derivative of X at p onto T_pM.

Theorem 41 (Gauss Formula) *Let U, V be vector fields on M which are arbitrarily extended to X, then the following holds along M*

$$\tilde{\nabla}_U V = \nabla_U V + \Pi(U,V),$$

where $\tilde{\nabla}$ is the covariant derivative of X and ∇ the covariant derivative of M.

The second fundamental form connects also the curvature tensors of X and M.

Theorem 42 (Gauss equation) *For any $U,V,W,Z \in T_pM$ the following equation holds*

$$\tilde{R}(U,V,W,Z) = R(U,V,W,Z) - \langle \Pi(U,Z), \Pi(V,W) \rangle + \langle \Pi(U,W), \Pi(V,Z) \rangle,$$

where $\tilde{R}$ and R are the Riemann curvature[8] *tensors of X and M.*

In this paper we derive a relationship between distances in M and the corresponding distances in X. Since Riemannian manifolds are length spaces and therefore the distance is induced by length minimizing curves (locally the geodesics), it is of special interest to connect properties of curves of M with respect to X. Applying the Gauss Formula to a curve $c(t) : (t_0, t_1) \to M$ yields the following

$$\tilde{D}_t V = D_t V + \Pi(V, \dot{c}),$$

where $\tilde{D}_t = \dot{c}^a \tilde{\nabla}_a$ and $\dot{c}$ is the tangent vector field to the curve $c(t)$. Now let $c(t)$ be a geodesic parameterized by arc-length, that is with unit-speed, then its acceleration fulfills $D_t \dot{c} = \dot{c}^a \nabla_a \dot{c}^b = 0$ (however that is only true locally in the interior of M, globally if M has boundary length minimizing curves may behave differently especially if a length minimizing curve goes along the boundary its acceleration can be non-zero), and one gets for the acceleration in the ambient space

$$\tilde{D}_t \dot{c} = \Pi(\dot{c}, \dot{c}).$$

In our setting where $X = \mathbb{R}^d$ the term $\tilde{D}_t \dot{c}$ is just the ordinary acceleration $\ddot{c}$ in $\mathbb{R}^d$. Remember that the norm of the acceleration vector is inverse to the curvature of the curve at that point (if c is parameterized by arc-length).[9] Due to this connection it becomes more apparent why the second fundamental form is often called the extrinsic curvature (with respect to X).

The following Lemma shows that the second fundamental form Π of an isometrically embedded submanifold M of $\mathbb{R}^d$ is in normal coordinates just the Hessian of i.

Lemma 43 *Let e_α, $\alpha = 1, \ldots, d$ denote an orthonormal basis of $T_{i(x)}\mathbb{R}^d$ then the second fundamental form of M in normal coordinates at y is given as:*

$$\Pi(\partial_{y^i}, \partial_{y^j})\Big|_0 = \frac{\partial^2 i^\alpha}{\partial y^i \partial y^j} e_\alpha.$$

Proof Let $\tilde{\nabla}$ be the flat connection of $\mathbb{R}^d$ and ∇ the connection of M. Then by Theorem 41, $\Pi(\partial_{y^i}, \partial_{y^j}) = \tilde{\nabla}_{i^* \partial_{y^i}}(i^* \partial_{y^j}) - \nabla_{\partial_{y^i}} \partial_{y^j} = \partial_{y^i}\left(\frac{\partial i^\alpha}{\partial_{y^j}}\right) e_\alpha = \frac{\partial^2 i^\alpha}{\partial y^i \partial y^j} e_\alpha$, where the second equality follows from the flatness of $\tilde{\nabla}$ and $\Gamma^i_{jk}\Big|_0 = 0$ in normal coordinates. ■

8. The Riemann curvature tensor of a Riemannian manifold M is defined as $R : T_pM \otimes T_pM \otimes T_pM \to T_p^*M$,

$$R(X,Y)Z = \nabla_X \nabla_Y Z - \nabla_Y \nabla_X Z - \nabla_{[X,Y]} Z.$$

In local coordinates x^i, $R_{ijk}{}^l \partial_l = R(\partial_i, \partial_j)\partial_k$ and $R_{ijkm} = g_{lm} R_{ijk}{}^l$.

9. Note that if c is parameterized by arc-length, $\dot{c}$ is tangent to M, that is in particular $\|\dot{c}\|_{T_xX} = \|\dot{c}\|_{T_xM}$

Appendix B. Proofs and Lemmas

The following lemmas are needed in the proof of Proposition 22.

Lemma 44 *If the kernel $k : \mathbb{R}_+ \to \mathbb{R}_+$ satisfies Assumptions 20, then*

$$\int_{\mathbb{R}^m} \frac{\partial k}{\partial x}(\|u\|^2) u^i u^j u^k u^l du = -\frac{1}{2} C_2 \left[\delta^{ij}\delta^{kl} + \delta^{ik}\delta^{jl} + \delta^{il}\delta^{jk}\right].$$

Proof Note first that for a function $f(\|u\|^2)$ one has $\frac{\partial f}{\partial \|u\|^2} = \frac{\partial f}{\partial u_i^2}$. The rest follows from partial integration.

$$\int\limits_{-\infty}^{\infty} \frac{\partial k}{\partial u^2}(u^2)\, u^2\, du = \int\limits_0^{\infty} \frac{\partial k}{\partial v}(v)\, \sqrt{v}\, dv = \left[k(v)\,\sqrt{v}\right]_0^{\infty} - \int\limits_0^{\infty} k(v)\, \frac{1}{2\sqrt{v}}\, dv = -\frac{1}{2}\int\limits_{-\infty}^{\infty} k(u^2) du,$$

where $\left[k(v)\,\sqrt{v}\right]_0^{\infty} = 0$ due to the boundedness and exponential decay of k.

In the same way one can derive, $\int_{-\infty}^{\infty} \frac{\partial k}{\partial u^2}(u^2)\, u^4\, du = -\frac{3}{2}\int_{-\infty}^{\infty} k(u^2) u^2\, du$. The result follows by noting that since k is an even function only integration over even powers of coordinates will be non-zero. ∎

Lemma 45 *Let k satisfy Assumption 20 and let V_{ijkl} be a given tensor. Assume now $\|z\|^2 \geq \|z\|^2 + V_{ijkl} z^i z^j z^k z^l + \beta(z)\,\|z\|^5 \geq \frac{1}{4}\|z\|^2$ on $B(0, r_{\min}) \subset \mathbb{R}^m$, where $\beta(z)$ is continuous and $\beta(z) \sim O(1)$ as $z \to 0$. Then there exists a constant C and a $h_0 > 0$ such that for all $h < h_0$ and all $f \in C^3(B(0, r_{\min}))$,*

$$\begin{aligned} &\Big| \int_{B(0,r_{\min})} k_h\left(\frac{\|z\|^2 + V_{ijkl} z^i z^j z^k z^l + \beta(z)\,\|z\|^5)}{h^2}\right) f(z) dz \\ &- \left(C_1 f(0) + C_2 \frac{h^2}{2}\left[(\Delta f)(0) - f(0)\sum_{i,k}^{m} V_{iikk} + V_{ikik} + V_{ikki}\right]\right)\Big| \leq C h^3, \end{aligned}$$

where C is a constant depending on k, $r_{\min}$, V_{ijkl} and $\|f\|_{C^3}$.

Proof As a first step we do a Taylor expansion of the kernel around $\|z\|^2/h^2$:

$$k_h\left(\frac{\|z\|^2 + \eta}{h^2}\right) = k_h\left(\frac{\|z\|^2}{h^2}\right) + \frac{\partial k_h}{\partial x}\Big|_{\frac{\|z\|^2}{h^2}} \frac{\eta}{h^2} + \frac{\partial^2 k_h(x)}{\partial x^2}\Big|_{\frac{\|z\|^2(1-\theta)+\theta\eta}{h^2}} \frac{\eta^2}{h^4},$$

where in the last term $0 \leq \theta(z) \leq 1$. We then decompose the integral:

$$\begin{aligned} &\int_{B(0,r_{\min})} k_h\left(\frac{\|z\|^2 + V_{ijkl} z^i z^j z^k z^l + \beta(z)\,\|z\|^5}{h^2}\right) f(z) dz \\ =&\int\limits_{\mathbb{R}^m}\left(k_h\left(\frac{\|z\|^2}{h^2}\right) + \frac{\partial k_h}{\partial x}\Big|_{\frac{\|z\|^2}{h^2}} \frac{V_{ijkl}\, z^i z^j z^k z^l}{h^2}\right)\left(f(0) + \left\langle \nabla f\big|_0, z\right\rangle + \frac{1}{2}\frac{\partial^2 f}{\partial z^i \partial z^j}\Big|_0 z^i z^j\right) dz + \sum_{i=0}^{4} \alpha_i, \end{aligned}$$

where we define the five error terms α_i as:

$$\alpha_0 = \int_{B(0,r_{\min})} \frac{\partial k_h}{\partial x}\Big|_{\frac{\|z\|^2}{h^2}} \frac{\beta(z)\,\|z\|^5}{h^2} f(z)\,dz,$$

$$\alpha_1 = \int_{B(0,r_{\min})} \frac{\partial^2 k_h}{\partial x^2}\Big|_{\frac{\|z\|^2(1-\theta)+\theta\eta}{h^2}} \frac{\left(V_{ijkl}z^iz^jz^kz^l+\beta(z)\,\|z\|^5\right)^2}{h^4} f(z)dz,$$

$$\alpha_2 = \int_{B(0,r_{\min})} k_h\left(\frac{\|z\|^2+V_{ijkl}z^iz^jz^kz^l+\beta(z)\,\|z\|^5}{h^2}\right)\frac{1}{6}\frac{\partial^3 f}{\partial z^i\partial z^j\partial z^k}(\theta z)z^iz^jz^k dz,$$

$$\alpha_3 = \int_{\mathbb{R}^m\setminus B(0,r_{\min})} k_h\left(\frac{\|z\|^2}{h^2}\right)\left(f(0)+\left\langle \nabla f\big|_0, z\right\rangle+\frac{1}{2}\frac{\partial^2 f}{\partial z^i\partial z^j}\Big|_0 z^iz^j\right)dz,$$

$$\alpha_4 = \int_{\mathbb{R}^m\setminus B(0,r_{\min})} \frac{\partial k_h}{\partial x}\Big|_{\frac{\|z\|^2}{h^2}} \frac{V_{ijkl}\,z^iz^jz^kz^l}{h^2}\left(f(0)+\left\langle \nabla f\big|_0, z\right\rangle+\frac{1}{2}\frac{\partial^2 f}{\partial z^i\partial z^j}\Big|_0 z^iz^j\right)dz,$$

where in α_1, $\eta = V_{ijkl}z^iz^jz^kz^l+\beta(z)\,\|z\|^5$. With $\int_{\mathbb{R}^m} k(\|z\|^2)\,z_i\,dz = 0,\ \forall i$, and $\int_{\mathbb{R}^m} k(\|z\|^2)\,z_i\,z_j dz = 0$ if $i \neq j$, and Lemma 44 the main term simplifies to:

$$\begin{aligned}
&\int_{\mathbb{R}^m}\left(k_h\left(\frac{\|z\|^2}{h^2}\right)+\frac{\partial k_h(x)}{\partial x}\Big|_{\frac{\|z\|^2}{h^2}}\frac{V_{ijkl}\,z^iz^jz^kz^l}{h^2}\right)\left(f(0)+\frac{1}{2}\frac{\partial^2 f}{\partial z^i\partial z^j}\Big|_0 z^iz^j\right)dz\\
=&\int_{\mathbb{R}^m}\left(k(\|u\|^2)+h^2\frac{\partial k(x)}{\partial x}\Big|_{\|u\|^2}V_{ijkl}\,u^iu^ju^ku^l\right)\left(f(0)+\frac{h^2}{2}\frac{\partial^2 f}{\partial z^i\partial z^j}\Big|_0 u^iu^j\right)du\\
=&C_1f(0)-\frac{h^2}{2}C_2f(0)V_{ijkl}\left[\delta^{ij}\delta^{kl}+\delta^{ik}\delta^{jl}+\delta^{il}\delta^{jk}\right]+\frac{h^2}{2}C_2\sum_{i=1}^m\frac{\partial^2 f}{\partial(z^i)^2}\Big|_0+O(h^4),
\end{aligned}$$

where the $O(h^4)$ term is finite due to the exponential decay of k and depends on k, $r_{\min}$, V_{ijkl} and $\|f\|_{C^3}$. Now we can upper bound the remaining error terms α_i, $i=0,\ldots,4$. For the argument of the kernel in α_1 and α_2 we have by our assumptions on $B(0,r_{\min})$:

$$\frac{\|z\|^2}{h^2} \geq \frac{\|z\|^2+V_{ijkl}z^iz^jz^kz^l+\beta(z)\,\|z\|^5}{h^2} \geq \frac{\|z\|^2}{4h^2}.$$

Note that this inequality implies that β is uniformly bounded on $B(0,r_{\min})$ in terms of $r_{\min}$ and V_{ijkl}. Moreover, for small enough h we have $\frac{r_{\min}}{h} \geq \sqrt{A}$ (see Assumptions 20 for the definition of A) so that we can use the exponential decay of k for α_3 and α_4,

$$|\alpha_0| \leq h^3\,\|f\|_{C^3}\int_{B(0,\frac{r_{\min}}{h})}\frac{\partial k_h}{\partial x}\Big|_{\|u\|^2}|\beta(hu)|\,\|u\|^5\,du.$$

Since $\frac{\partial k_h}{\partial x}$ is bounded and has exponential decay, one has $|\alpha_0| \leq K_0\,h^3$ where K_0 depends on k, $r_{\min}$ and $\|f\|_{C^3}$.

$$\begin{aligned}
|\alpha_1| \leq &\int_{B(0,r_{\min})}\left|\frac{\partial^2 k_h}{\partial x^2}\left(\frac{\|z\|^2(1-\theta)+\theta\eta}{h^2}\right)\right|\left|\frac{\left(V_{ijkl}z^iz^jz^kz^l+\beta(z)\,\|z\|^5\right)^2}{h^4}f(z)dz\right.\\
\leq &h^4\,\|f\|_{C^3}\int_{B(0,\frac{r_{\min}}{h})}\left|\frac{\partial^2 k}{\partial x^2}\left(\|u\|^2(1-\theta)+\theta\eta\right)\right|\left(m^2\max_{i,j,k,l}|V_{ijkl}|\,\|u\|^4+h\,\|\beta\|_\infty\,\|u\|^5\right)^2du.
\end{aligned}$$

First suppose $\frac{r_{\min}}{h} \le 2\sqrt{A}$ then the integral is bounded since the integrands are bounded on $B(0, \frac{r_{\min}}{h})$. Now suppose $\frac{r_{\min}}{h} \ge 2\sqrt{A}$ and decompose $B(0, \frac{r_{\min}}{h})$ as $B(0, \frac{r_{\min}}{h}) = B(0, 2\sqrt{A}) \cup B(0, \frac{r_{\min}}{h}) \backslash B(0, 2\sqrt{A})$. On $B(0, 2\sqrt{A})$ the integral is finite since $\left|\frac{\partial^2 k}{\partial x^2}\right|$ is bounded and on the complement the integral is also finite since $\left|\frac{\partial^2 k}{\partial x^2}\right|$ has exponential decay since by assumption

$$\|u\|^2(1-\theta(hu)) + \theta(hu)\eta(hu) \ge \frac{1}{4}\|u\|^2 \ge A.$$

Therefore there exists a constant K_1 such that $|\alpha_1| \le K_1 h^4$. Moreover,

$$|\alpha_2| \le \int_{B(0,r_{\min})} k_h\left(\frac{\|z\|^2}{4h^2}\right)\frac{1}{6}\frac{\partial^3 f}{\partial z^i \partial z^j \partial z^k}(\theta z) z^i z^j z^k dz \le \frac{m^{3/2}\|f\|_{C^3} h^3}{6}\int_{\mathbb{R}^m} k\left(\frac{\|u\|^2}{4}\right)\|u\|^3 du \le K_2 h^3,$$

$$\begin{aligned}
|\alpha_3| &\le \int_{\mathbb{R}^m \backslash B(0,r_{\min})} k_h\left(\frac{\|z\|^2}{h^2}\right)\left(f(0) + \left\langle \nabla f\big|_0, z\right\rangle + \frac{1}{2}\frac{\partial^2 f}{\partial z^i \partial z^j}\Big|_0 z^i z^j\right) dz \\
&\le c\,\|f\|_{C^3} \int_{\mathbb{R}^m \backslash B(0,r_{\min})} e^{-\frac{\alpha\|z\|^2}{h^2}}(1 + m h^2\|z\|^2) dz \le c\, e^{-\alpha\frac{r_{\min}^2}{2h^2}}\left(\frac{2\pi}{\alpha}\right)^{\frac{m}{2}}\left(1 + m h^2 \frac{m}{\alpha}\right), \\
|\alpha_4| &\le K_4 \int_{\mathbb{R}^m \backslash B(0,r_{\min})} \left|\frac{\partial k_h}{\partial x}\left(\frac{\|z\|^2}{h^2}\right)\right| \frac{\|z\|^4 + \|z\|^6}{h^2} dz \le c K_4 h^2 e^{-\alpha\frac{r_{\min}^2}{2h^2}} \int_{\mathbb{R}^m} e^{-\alpha\|u\|^2}(\|u\|^4 + h^2\|u\|^6) du,
\end{aligned}$$

where K_4 is a constant depending on $\max_{i,j,k,l}|V_{ijkl}|$ and $\|f\|_{C^3}$. Now one has[10]: $e^{-\frac{\xi^2}{h^2}} \le h^s/\xi^s$ for $h \le \xi/s$. In particular, it holds $h^3 \ge e^{-\alpha\frac{r_{\min}^2}{2h^2}}$ for $h \le \frac{1}{3}\sqrt{\frac{\alpha}{2}}\, r_{\min}$, so that for $h < \min\{\frac{1}{3}\sqrt{\frac{\alpha}{2}}\, r_{\min}, \frac{r_{\min}}{\sqrt{A}}\} = h_0$ all error terms are smaller than a constant times h^3 where the constant depends on k, $r_{\min}$, V_{ijkl} and $\|f\|_{C^3}$. This finishes the proof. ■

Now we are ready to prove Proposition 22,

Proof Let $\varepsilon = \frac{1}{3}\min\{\mathrm{inj}(x), \pi\rho\}$[11] where ε is positive by the assumptions on M. Then we decompose M as $M = B(x,\varepsilon) \cup (M \backslash B(x,\varepsilon))$ and integrate separately. The integral over $M \backslash B(x,\varepsilon)$ can be upper bounded by using the definition of $\delta(x)$ (see Assumption 19) and the fact that k is non-increasing:

$$\begin{aligned}
\int_M k_h\big(\|i(x) - i(y)\|^2_{\mathbb{R}^d}\big) f(y) p(y) \sqrt{\det g}\, dy = &\int_{B(x,\varepsilon)} k_h\big(\|i(x) - i(y)\|^2_{\mathbb{R}^d}\big) f(y) p(y) \sqrt{\det g}\, dy \\
&+ \int_{M \backslash B(x,\varepsilon)} k_h\big(\|i(x) - i(y)\|^2_{\mathbb{R}^d}\big) f(y) p(y) \sqrt{\det g}\, dy.
\end{aligned}$$

Since k is non-increasing, we have the following inequality for the integral over $M \backslash B(x,\varepsilon)$:

$$\int_{M \backslash B(x,\varepsilon)} k_h\left(\|i(x) - i(y)\|^2_{\mathbb{R}^d}\right) f(y) p(y) \sqrt{\det g}\, dy \le \frac{1}{h^m} k\left(\frac{\delta(x)^2}{h^2}\right)\|f\|_\infty.$$

10. This inequality can be deduced from $e^x \ge x^n$ for all $x \ge 4n^2$.

11. The factor $1/3$ is needed in Theorem 25.

Since $\delta(x)$ is positive by assumption and k decays exponentially, we can make the upper bound smaller than h^3 for small enough h. Now we deal with the integral over $B(x,\varepsilon)$. Since ε is smaller than the injectivity radius $\mathrm{inj}(x)$, we can introduce normal coordinates $z = \exp_x^{-1}(y)$ on $B(x,\varepsilon)$, so that we can rewrite the integral using Proposition 15 as:

$$\int_{B(0,\varepsilon)} k_h\left(\frac{\|z\|^2 - \frac{1}{12}\sum_{\alpha=1}^{d}\frac{\partial^2 i^\alpha}{\partial z^a \partial z^b}\frac{\partial^2 i^\alpha}{\partial z^u \partial z^v} z^a z^b z^u z^v + O(\|z\|^5)}{h^2}\right) p(z) f(z)\sqrt{\det g}\, dz. \tag{5}$$

Using our assumptions, we see that $pf\sqrt{\det g}$ is in $C^3(B(0,\varepsilon))$. Moreover, by Corollary 17 one has for $d_M(x,y) \leq \pi\rho$, $\frac{1}{2}d_M(x,y) \leq \|x-y\| \leq d_M(x,y)$. Therefore we can apply Lemma 45 and compute the integral in (5) which results in:

$$p(0)f(0)\left(C_1 + \frac{h^2 C_2}{24}\sum_{\alpha=1}^{d}\frac{\partial^2 i^\alpha}{\partial z^a \partial z^b}\frac{\partial^2 i^\alpha}{\partial z^c \partial z^d}\left[\delta^{ab}\delta^{cd} + \delta^{ac}\delta^{bd} + \delta^{ad}\delta^{bc}\right]\right)$$
$$+\frac{h^2 C_2}{2}\Delta_M(pf\sqrt{\det g})\Big|_0 + O(h^3), \tag{6}$$

where we have used that in normal coordinates z^i at 0 the Laplace-Beltrami operator Δ_M is given as $\Delta_M f\Big|_x = \sum_{i=1}^m \frac{\partial^2 f}{\partial (z^i)^2}\Big|_0$. The second term in the above equation can be evaluated using the Gauss equations, see Smolyanov, von Weizsäcker, and Wittich (2007, Proposition 6).

$$\begin{aligned}
\sum_{a,b=1}^{m}\sum_{\alpha=1}^{d}\frac{\partial^2 i^\alpha}{\partial z^a \partial z^b}\frac{\partial^2 i^\alpha}{\partial z^c \partial z^d}\left[\delta^{ab}\delta^{cd} + \delta^{ac}\delta^{bd} + \delta^{ad}\delta^{bc}\right] &= \sum_{a,b=1}^{m}\sum_{\alpha=1}^{d}\frac{\partial^2 i^\alpha}{\partial (z^a)^2}\frac{\partial^2 i^\alpha}{\partial (z^b)^2} + 2\frac{\partial^2 i^\alpha}{\partial z^a \partial z^b}\frac{\partial^2 i^\alpha}{\partial z^a \partial z^b}\\
&= 2\sum_{a,b=1}^{m}\sum_{\alpha=1}^{d}\left(\frac{\partial^2 i^\alpha}{\partial z^a \partial z^b}\frac{\partial^2 i^\alpha}{\partial z^a \partial z^b} - \frac{\partial^2 i^\alpha}{\partial (z^a)^2}\frac{\partial^2 i^\alpha}{\partial (z^b)^2}\right) + 3\sum_{a,b=1}^{m}\sum_{\alpha=1}^{d}\frac{\partial^2 i^\alpha}{\partial (z^a)^2}\frac{\partial^2 i^\alpha}{\partial (z^b)^2}\\
&= 2\sum_{a,b=1}^{m}\langle \Pi(\partial_{z^a},\partial_{z^b}), \Pi(\partial_{z^a},\partial_{z^b})\rangle - \langle \Pi(\partial_{z^a},\partial_{z^a}), \Pi(\partial_{z^b},\partial_{z^b})\rangle + 3\left\|\sum_{a=1}^{m}\Pi(\partial_{z^a},\partial_{z^a})\right\|^2_{T_{i(x)}\mathbb{R}^d}\\
&= -2R + 3\left\|\sum_{j=1}^{m}\Pi(\partial_{z^j},\partial_{z^j})\right\|^2_{T_{i(x)}\mathbb{R}^d},
\end{aligned}$$

where R is the scalar curvature and we used Lemma 43 in the third equality. Plugging this result into (6) and using from Proposition 15, $\Delta_M\sqrt{\det g}\big|_0 = -\frac{1}{3}R$, we are done. ■

Lemma 46 *Let k have compact support on $[0, R_k^2]$ and let $0 < h \leq h_{\max}$. Then for any $x \in M$ there exist constants $D_1, D_2 > 0$ independent of h such that for any $y \in B_{\mathbb{R}^d}(x, hR_k) \cap M$,*

$$0 < D_1 \leq p_h(y) \leq D_2.$$

Proof First suppose that $hR_k < s := \min\{\kappa/2, R_0/2\}$. Since $\|y - z\| \leq hR_k \leq \kappa/2$ we have by Lemma 18: $\frac{1}{2}d_M(y,z) \leq \|y-z\| \leq d_M(y,z)$. Moreover, since $p(x) > 0$ on M and p is bounded and continuous, there exist lower and upper bounds $p_{\min}$ and $p_{\max}$ on the density on $B_M(x, 4hR_k)$. That implies

$$p_h(y) \leq \frac{\|k\|_\infty}{h^m} p_{\max}\int_{B_M(y,2hR_k)}\sqrt{\det g}\, dz \leq \|k\|_\infty\, p_{\max}\, S_2\, 2^m R_k^m,$$

where the last inequality follows from Lemma 14. Note further that $d_M(x,y) \leq 2hR_k$ and $d_M(y,z) \leq 2hR_k$ implies $d_M(x,z) \leq 4hR_k$. Since the kernel function is continuous there exists an r_k such that $k(x) \geq \|k\|_\infty /2$ for $0 < x \leq r_k$. We get

$$p_h(y) \geq \frac{\|k\|_\infty}{2h^m} \int\limits_{B_{\mathbb{R}^d}(x,hr_k)\cap M} p(z)\sqrt{\det g}\, dz \geq \frac{\|k\|_\infty}{2h^m} p_{\min} \operatorname{vol}_M(B_M(x,hr_k)) \geq \frac{\|k\|_\infty}{2} p_{\min} S_1 r_k^m.$$

Now suppose $s \leq hR_k$ and $h \leq h_{\max}$. Then $p_h(y) \leq \frac{\|k\|_\infty}{h^m} \leq \|k\|_\infty \left(\frac{R_k}{s}\right)^m$. For the lower bound we get

$$\begin{aligned} p_h(y) &\geq \int_M k_h(d_M(y,z))p(z)\sqrt{\det g}\, dz \geq \int_{B_M(y,hr_k)} k_h(d_M(y,z))p(z)\sqrt{\det g}\, dz \\ &\geq \frac{\|k\|_\infty}{2h^m} \mathrm{P}\Big(B_M(y,hr_k)\Big) \geq \frac{\|k\|_\infty}{2h^m_{\max}} \mathrm{P}\Big(B_M(y,s\frac{r_k}{R_k})\Big). \end{aligned}$$

Since p is continuous and $p > 0$, the function $y \to \mathrm{P}\Big(B_M(y,s\frac{r_k}{R_k})\Big)$ is continuous and positive and therefore has a lower bound greater zero on the ball $B_{\mathbb{R}^d}(x,hR_k)\cap M$. ■

References

R. Alexander and S. Alexander. Geodesics in Riemannian manifolds with boundary. *Indiana Univ. Math. J.*, 30:481–488, 1981.

M. Belkin. *Problems of Learning on Manifolds*. PhD thesis, University of Chicago, 2003. http://www.people.cs.uchicago.edu /˜misha /thesis.pdf.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comp.*, 15(6):1373–1396, 2003.

M. Belkin and P. Niyogi. Semi-supervised learning on manifolds. *Machine Learning*, 56:209–239, 2004.

M. Belkin and P. Niyogi. Towards a theoretical foundation for Laplacian-based manifold methods. In P. Auer and R. Meir, editors, *Proc. of the 18th Conf. on Learning Theory (COLT)*, pages 470–485, Berlin, 2005. Springer.

P. H. Bérard. *Spectral Geometry: Direct and Inverse Problems*. Springer, Berlin, 1986.

M. Bernstein, V. de Silva, J. C. Langford, and J.B. Tenenbaum. Graph approximations to geodesics on embedded manifolds. Technical report, Stanford University, 2001.

O. Bousquet, O. Chapelle, and M. Hein. Measure based regularization. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, volume 16. MIT Press, 2004.

S. Canu and A. Elisseeff. Regularization, kernels and sigmoid net. unpublished, 1999.

F. Chung. *Spectral Graph Theory*. AMS, Providence, RI, 1997.

F. Chung and R. P. Langlands. A combinatorial Laplacian with vertex weights. *J. Combinatorial Th. Ser. A*, 75:316–327, 1996.

S. Coifman and S. Lafon. Diffusion maps. *Appl. Comput. Harmon. Anal.*, 21:5–30, 2006.

J. Dodziuk. Difference equation, isoperimetric inequality and transience of certain random walks. *Trans. Am. Math. Soc.*, 284:787–794, 1984.

E. Giné and V. Koltchinskii. Empirical graph Laplacian approximation of Laplace-Beltrami operators: Large sample results. *High Dimensional Probability*, 51:238–259, 2006.

W. Greblicki, A. Krzyzak, and M. Pawlak. Distribution-free pointwise consistency of kernel regression estimate. *Ann. Stat.*, 12:1570–1575, 1984.

A. Grigoryan. Heat kernels on weighted manifolds and applications. *Cont. Math.*, 398:93–191, 2006.

L. Györfi, M. Kohler, A. Krzyżak, and H. Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer, New York, 2004.

M. Hein. *Geometrical aspects of statistical learning theory*. PhD thesis, MPI für biologische Kybernetik/Technische Universität Darmstadt, 2005.

M. Hein. Uniform convergence of adaptive graph-based regularization. In G. Lugosi and H. Simon, editors, *Proc. of the 19th Conf. on Learning Theory (COLT)*, pages 50–64, Berlin, 2006. Springer.

M. Hein and J.-Y. Audibert. Intrinsic dimensionality estimation of submanifolds in $\mathbb{R}^d$. In L. De Raedt and S. Wrobel, editors, *Proc. of the 22nd Int. Conf. on Machine Learning (ICML)*, pages 289–296, 2005.

M. Hein, J.-Y. Audibert, and U. von Luxburg. From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians. In P. Auer and R. Meir, editors, *Proc. of the 18th Conf. on Learning Theory (COLT)*, pages 486–500, Berlin, 2005. Springer.

J. Jost. *Riemannian Geometry and Geometric Analysis*. Springer-Verlag, Berlin, third edition, 2002.

W. Klingenberg. *Riemannian Geometry*. De Gruyter, Berlin, 1982.

S. S. Lafon. *Diffusion Maps and Geometric Harmonics*. PhD thesis, Yale University, 2004. http://www.math.yale.edu /~sl349 /publications /dissertation.pdf.

J. M. Lee. *Riemannian Manifolds*. Springer, New York, 1997.

J. M. Lee. *Introduction to Smooth Manifolds*. Springer, New York, 2003.

P. McDonald and R. Meyers. Diffusions on graphs, Poisson problems and spectral geometry. *Trans. Amer. Math. Soc.*, 354:5111–5136, 2002.

S. Rosenberg. *The Laplacian on a Riemannian Manifold*. Cambridge University Press, 1997.

T. Schick. *Analysis on ∂-manifolds of Bounded Geometry, Hodge-De Rham Isomorphsim and L^2-Index Theorem*. PhD thesis, Universität Mainz, 1996.

T. Schick. Manifolds with boundary of bounded geometry. *Math. Nachr.*, 223:103–120, 2001.

A. Singer. From graph to manifold Laplacian: The convergence rate. *Appl. Comput. Harmon. Anal.*, 21:128–134, 2006.

O. G. Smolyanov, H. von Weizsäcker, and O. Wittich. Brownian motion on a manifold as limit of stepwise conditioned standard Brownian motions. In *Stochastic Processes, Physics and Geometry: New Interplays, II*, volume 29, pages 589–602, Providence, RI, 2000. AMS.

O. G. Smolyanov, H. von Weizsäcker, and O. Wittich. Chernoff's theorem and discrete time approximations of Brownian motion on manifolds. *Potential Analysis*, 26:1–29, 2007.

O. Sorkine. Differential representations for mesh processing. *Computer Graphics Forum*, 25(4): 789–807, 2006.

D. Spielman and S. Teng. Spectral partioning works: planar graphs and finite element meshes. In *37th Ann. Symp. on Found. of Comp. Science (FOCS)*, pages 96–105. IEEE Comp. Soc. Press, 1996.

N. T. Varopoulos. Brownian motion and random walks on manifolds. *Ann. Inst. Fourier*, 34:243–269, 1984.

U. von Luxburg. A tutorial on spectral clustering. Technical Report 149, Max Planck Institute for Biological Cybernetics, 2006.

U. von Luxburg, M. Belkin, and O. Bousquet. Consistency of spectral clustering, 2007. to appear in Ann. Stat.

W. Woess. *Random Walks on Infinite Graphs and Groups*. Cambridge University Press, Cambridge, 2000.

G. Xu. Discrete Laplace-Beltrami operators and their convergence. *Computer Aided Geometric Design*, 21:767–784, 2004.

D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, volume 16, pages 321–328. MIT Press, 2004.

D. Zhou, B. Schölkopf, and T. Hofmann. Semi-supervised learning on directed graphs. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, volume 17, pages 1633–1640. MIT Press, 2005.

X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical Report CMU-CALD-02-107, 2002.

X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In T. Fawcett and N. Mishra, editors, *Proc. of the 20nd Int. Conf. on Machine Learning (ICML)*, 2003.

Journal of Machine Learning Research 8 (2007) 1369-1392 Submitted 4/06; Revised 1/07; Published 7/07

Generalization Error Bounds in Semi-supervised Classification Under the Cluster Assumption

Philippe Rigollet RIGOLLET@MATH.GATECH.EDU
School of Mathematics
Georgia Institute of Technology
Atlanta, GA 30332-0160, U.S.A

Editor: Gábor Lugosi

Abstract

We consider semi-supervised classification when part of the available data is unlabeled. These unlabeled data can be useful for the classification problem when we make an assumption relating the behavior of the regression function to that of the marginal distribution. Seeger (2000) proposed the well-known *cluster assumption* as a reasonable one. We propose a mathematical formulation of this assumption and a method based on density level sets estimation that takes advantage of it to achieve fast rates of convergence both in the number of unlabeled examples and the number of labeled examples.

Keywords: semi-supervised learning, statistical learning theory, classification, cluster assumption, generalization bounds

1. Introduction

Semi-supervised classification has been of growing interest over the past few years and many methods have been proposed. The methods try to give an answer to the question: "How to improve classification accuracy using unlabeled data together with the labeled data?". Unlabeled data can be used in different ways depending on the assumptions on the model. There are mainly two approaches to solve this problem. The first one consists in using the unlabeled data to reduce the *complexity* of the problem in a broad sense. For instance, assume that we have a set of potential classifiers and we want to aggregate them. In that case, unlabeled data is used to measure the *compatibility* between the classifiers and reduces the complexity of the set of candidate classifiers (see, for example, Balcan and Blum, 2005; Blum and Mitchell, 1998). Unlabeled data can also be used to reduce the dimension of the problem, which is another way to reduce complexity. For example, in Belkin and Niyogi (2004), it is assumed that the data actually live on a submanifold of low dimension.

The second approach is the one that we use here. It assumes that the data contains clusters that have homogeneous labels and the unlabeled observations are used to identify these clusters. This is the so-called *cluster assumption*. This idea can be put in practice in several ways giving rise to various methods. The simplest is the one presented here: estimate the clusters, then label each cluster uniformly. Most of these methods use Hartigan's (1975) definition of clusters, namely the connected components of the density level sets. However, they use a parametric—usually mixture—model to estimate the underlying density which can be far from reality. Moreover, no generalization error bounds are available for such methods. In the same spirit, Tipping (1999) and Rattray (2000) propose methods that learn a distance using unlabeled data in order to have intra-cluster distances smaller than inter-clusters distances. The whole family of graph-based methods aims also at using unlabeled data to learn the distances between points. The edges of the graphs reflect the proximity between points. For a detailed survey on graph methods we refer to Zhu (2005). Finally, we mention kernel methods, where unlabeled data are used to build the kernel. Recalling that the kernel measures proximity between points, such methods can also be viewed as learning a distance using unlabeled data (see Bousquet et al., 2004; Chapelle and Zien, 2005; Chapelle et al., 2006).

The cluster assumption can be interpreted in another way, that is, as the requirement that the decision boundary has to lie in low density regions. This interpretation has been widely used in learning since it can be used in the design of standard algorithms such as Boosting (d'Alché Buc et al., 2001; Hertz et al., 2004) or SVM (Bousquet et al., 2004; Chapelle and Zien, 2005), which are closely related to kernel methods mentioned above. In these algorithms, a greater penalization is given to decision boundaries that cross a cluster. For more details, see, for example, Seeger (2000), Zhu (2005), and Chapelle et al. (2006). Although most methods make, sometimes implicitly, the cluster assumption, no formulation in probabilistic terms has been provided so far. The formulation that we propose in this paper remains very close to its original text formulation and allows to derive generalization error bounds. We also discuss what can and cannot be done using unlabeled data. One of the conclusions is that considering the whole excess-risk is too ambitious and we need to concentrate on a smaller part of it to observe the improvement of semi-supervised classification over supervised classification.

1.1 Outline of the Paper

After describing the model, we formulate the cluster assumption and discuss why and how it can improve classification performance in Section 2. The main result of this section is Proposition 2.1 which essentially states that the effect of unlabeled data on the rates of convergence cannot be observed on the whole excess-risk. We therefore introduce the *cluster excess-risk* which corresponds to a part of the excess-risk that is interesting for this problem. In Section 3, we study the population case where the clusters are perfectly known, to get an idea of our target. Indeed, such a population case corresponds in some way to the case where the amount of unlabeled data is infinite. Section 4 contains the main result: after having defined the clusters in terms of density level sets, we propose an algorithm for which we derive rates of convergence for the cluster excess-risk as a measure of performance. An example of consistent density level set estimators is given in Section 5. Section 6 is devoted to a discussion on the choice of the level as well as possible implementations and improvements. Proofs of the results are gathered in Section A.

1.2 Notation

Throughout the paper, we denote positive constants by c_j. We write Γ^c for the complement of the set Γ. For two sequences $(u_p)_p$ and $(v_p)_p$ (in that paper, p will be m or n), we write $u_p = O(v_p)$ if there exists a constant $C > 0$ such that $u_p \leq C v_p$ and we write $u_p = \widetilde{O}(v_p)$ if $u_p \leq C(\log p)^{\alpha} v_p$ for some constants $\alpha > 0, C > 0$. Moreover, we write $u_p = o(v_p)$, if there exists a non negative sequence $(\varepsilon_p)_p$ that tends to 0 when p tends to infinity and such that $|u_p| \leq \varepsilon_p |v_p|$. Thus, if $u_p = \widetilde{O}(v_p)$, we have $u_p = o(v_p p^{\beta})$, for any $\beta > 0$.

2. The Model

Let (X,Y) be a random couple with joint distribution P, where $X \in \mathcal{X} \subset \mathbb{R}^d$ is a vector of d features and $Y \in \{0,1\}$ is a label indicating the class to which X belongs. The distribution P of the random couple (X,Y) is completely determined by the pair (P_X, η) where P_X is the marginal distribution of X and η is the regression function of Y on X, that is, $\eta(x) \triangleq P(Y=1|X=x)$. The goal of classification is to predict the label Y given the value of X, that is, to construct a measurable function $g : \mathcal{X} \to \{0,1\}$ called a *classifier*. The performance of g is measured by the average classification error

$$R(g) \triangleq P(g(X) \neq Y) .$$

A minimizer of the risk $R(g)$ over all classifiers is given by the *Bayes classifier* $g^\star(x) = \mathbb{I}_{\{\eta(x) \geq 1/2\}}$, where $\mathbb{I}_{\{\cdot\}}$ denotes the indicator function. Assume that we have a sample of n observations (X_1,Y_1), $\ldots,(X_n,Y_n)$ that are independent copies of (X,Y). An empirical classifier is a random function $\hat{g}_n : \mathcal{X} \to \{0,1\}$ constructed on the basis of the sample $(X_1,Y_1),\ldots,(X_n,Y_n)$. Since $g^\star$ is the best possible classifier, we measure the performance of an empirical classifier $\hat{g}_n$ by its *excess-risk*

$$\mathcal{E}(\hat{g}_n) = \mathbb{E}_n R(\hat{g}_n) - R(g^\star) ,$$

where $\mathbb{E}_n$ denotes the expectation with respect to the joint distribution of the sample $(X_1,Y_1),\ldots,$ (X_n,Y_n). We denote hereafter by $\mathbb{P}_n$ the corresponding probability.

In many applications, a large amount of unlabeled data is available together with a small set of labeled data $(X_1,Y_1),\ldots,(X_n,Y_n)$ and the goal of semi-supervised classification is to use unlabeled data to improve the performance of classifiers. Thus, we observe two independent samples $\mathbb{X}_l = \{(X_1,Y_1),\ldots,(X_n,Y_n)\}$ and $\mathbb{X}_u = \{X_{n+1},\ldots,X_{n+m}\}$, where n is rather small and typically $m \gg n$. Most existing theoretical studies of supervised classification use empirical processes theory (Devroye et al., 1996; Vapnik, 1998; van de Geer, 2000; Boucheron et al., 2005) to obtain rates of convergence for the excess-risk that are polynomial in n. Typically these rates are of the order $O(1/\sqrt{n})$ and can be as small as $\widetilde{O}(1/n)$ under some low noise assumptions (cf. Tsybakov, 2004; Audibert and Tsybakov, 2007). However, simulations indicate that much faster rates should be attainable when unlabeled data is used to identify homogeneous clusters. Of course, it is well known that in order to make use of the additional unlabeled observations, we have to make an assumption on the dependence between the marginal distribution of X and the joint distribution of (X,Y) (see, for example, Zhang and Oles, 2000). Seeger (2000) formulated the rather intuitive *cluster assumption* as follows[1]

> Two points $x, x' \in \mathcal{X}$ should have the same label y if there is a path between them which passes only through regions of relatively high P_X.

This assumption, in its raw formulation cannot be exploited in the probabilistic model since (i) the labels are random variables Y, Y' so that the expression "should have the same label" is meaningless unless η takes values in $\{0,1\}$ and (ii) it is not clear what "regions of relatively high P_X" are. To match the probabilistic framework, we propose the following modifications.

(i) Assume $P[Y = Y'|X, X' \in C] \geq P[Y \neq Y'|X, X' \in C]$, where C is a cluster.

(ii) Define "regions of relatively high P_X" in terms of *density level sets*.

1. The notation is adapted to the present framework.

Assume for the moment that we know what the clusters are, so that we do not have to define them in terms of density level sets. This will be done in Section 4. Let $T_1, T_2, \ldots,$ be a countable family of subsets of $\mathcal{X}$. We now make the assumption that the T_j's are clusters of homogeneous data.

Cluster Assumption (CA) Let $T_1, T_2, \ldots,$ be a collection of measurable sets (clusters) such that $T_j \subset \mathcal{X}, j = 1, 2, \ldots$ Then the function $x \in \mathcal{X} \mapsto \mathbb{I}\{\eta(x) \geq 1/2\}$ takes a constant value on each of the $T_j, j = 1, 2, \ldots$.

It is not hard to see that the cluster assumption **(CA)** is equivalent to the following assumption.

Let $T_j, j = 1, 2, \ldots,$ be a collection of measurable sets such that $T_j \subset \mathcal{X},\ j = 1, 2, \ldots$ Then, for any $j = 1, 2, \ldots,$ we have

$$P[Y = Y' | X, X' \in T_j] \geq P[Y \neq Y' | X, X' \in T_j]\,.$$

A question remains: what happens outside of the clusters? Define the union of the clusters,

$$\mathcal{C} = \bigcup_{j \geq 1} T_j \tag{1}$$

and assume that we are in the problematic case, $P_X(\mathcal{C}^c) > 0$ such that the question makes sense. Since the cluster assumption **(CA)** says nothing about what happens outside of the set $\mathcal{C}$, we can only perform supervised classification on $\mathcal{C}^c$. Consider a classifier $\hat{g}_{n,m}$ built from labeled and unlabeled samples $(\mathbb{X}_l, \mathbb{X}_u)$ pooled together. The excess-risk of $\hat{g}_{n,m}$ can be written (see Devroye et al., 1996),

$$\mathcal{E}(\hat{g}_{n,m}) = \mathbb{E}_{n,m} \int_{\mathcal{X}} |2\eta(x) - 1| \mathbb{I}_{\{\hat{g}_{n,m}(x) \neq g^\star(x)\}} \mathrm{d}P_X(x)\,,$$

where $\mathbb{E}_{n,m}$ denotes the expectation with respect to the pooled sample $(\mathbb{X}_l, \mathbb{X}_u)$. We denote hereafter by $\mathbb{P}_{n,m}$ the corresponding probability. Since, the unlabeled sample is of no help to classify points in $\mathcal{C}^c$, any reasonable classifier should be based on the sample $\mathbb{X}_l$ so that $\hat{g}_{n,m}(x) = \hat{g}_n(x),\ \forall x \in \mathcal{C}^c$, and we have

$$\mathcal{E}(\hat{g}_{n,m}) \geq \mathbb{E}_n \int_{\mathcal{C}^c} |2\eta(x) - 1| \mathbb{I}_{\{\hat{g}_n(x) \neq g^\star(x)\}} \mathrm{d}P_X(x)\,. \tag{2}$$

Since we assumed $P_X(\mathcal{C}^c) \neq 0$, the RHS of (2) is bounded from below by the optimal rates of convergence that appear in supervised classification.

The previous heuristics can be stated more formally as follows. Recall that the distribution P of the random couple (X, Y) is completely characterized by the couple (P_X, η) where P_X is the marginal distribution of X and η is the regression function of Y on X. In the following proposition, we are interested in a class of distributions with cylinder form, that is, a class $\mathcal{D}$ that can be decomposed as $\mathcal{D} = \mathcal{M} \times \Xi$ where $\mathcal{M}$ is a fixed class of marginal distributions on $\mathcal{X}$ and Ξ is a fixed class of regression functions on $\mathcal{X}$ with values in $[0, 1]$.

Proposition 2.1 *Fix $n, m \geq 1$ and let $\mathcal{C}$ be a measurable subset of $\mathcal{X}$. Let $\mathcal{M}$ be a class of marginal distributions on $\mathcal{X}$ and let Ξ be a class of regression functions. Define the class of distributions $\mathcal{D}$ as $\mathcal{D} = \mathcal{M} \times \Xi$. Then, for any marginal distribution $P_X^0 \in \mathcal{M}$, we have*

$$\inf_{T_n} \sup_{\eta \in \Xi} \mathbb{E}_n \int_{\mathcal{C}^c} |2\eta - 1| \mathbb{I}_{\{T_n \neq g^\star\}} \mathrm{d}P_X^0 \leq \inf_{T_{n,m}} \sup_{P \in \mathcal{D}} \mathbb{E}_{n,m} \int_{\mathcal{C}^c} |2\eta - 1| \mathbb{I}_{\{T_{n,m} \neq g^\star\}} \mathrm{d}P_X\,, \tag{3}$$

where $\inf_{T_{n,m}}$ denotes the infimum over all classifiers based on the pooled sample $(\mathbb{X}_l, \mathbb{X}_u)$ and $\inf_{T_n}$ denotes the infimum over all classifiers based only on the labeled sample $\mathbb{X}_l$.

The main consequence of Proposition 2.1 is that even when the cluster assumption **(CA)** is valid the unlabeled data are useless to improve the rates of convergence. If the class $\mathcal{M}$ is reasonably large and satisfies $P_X^0(\mathcal{C}^c) > 0$, the left hand side in (3) can be bounded from below by the minimax rate of convergence with respect to n, over the class $\mathcal{D}$. Indeed a careful check of the proofs of minimax lower bounds reveals that they are constructed using a single marginal P_X^0 that is well chosen. These rates are typically of the order $n^{-\alpha}, 0 < \alpha \leq 1$ (see, for example, Mammen and Tsybakov 1999, Tsybakov 2004, and Audibert and Tsybakov 2007 and Boucheron et al. 2005 for a comprehensive survey).

Thus, unlabeled data do not improve the rate of convergence of this part of the excess-risk. To observe the effect of unlabeled data on the rates of convergence, we have to consider the *cluster excess-risk* of a classifier $\hat{g}_{n,m}$ defined by

$$\mathcal{E}_{\mathcal{C}}(\hat{g}_{n,m}) \triangleq \mathbb{E}_{n,m} \int_{\mathcal{C}} |2\eta(x) - 1| \mathbb{1}_{\{\hat{g}_{n,m}(x) \neq g^\star(x)\}} \mathrm{d}P_X(x) .$$

We will therefore focus on this measure of performance. The cluster excess-risk can also be expressed in terms of an excess-risk. To observe it, define the set $\mathcal{G}_{\mathcal{C}}$ of all classifiers restricted to $\mathcal{C}$:

$$\mathcal{G}_{\mathcal{C}} = \big\{ g : \mathcal{C} \to \{0,1\},\ g \text{ measurable} \big\} .$$

The performance of a classifier $g \in \mathcal{G}_{\mathcal{C}}$ is measured by the average classification error on $\mathcal{C}$:

$$R(g) = P\big(g(X) \neq Y\big) = P\big(g(X) \neq Y, X \in \mathcal{C}\big).$$

A minimizer of $R(\cdot)$ over $\mathcal{G}_{\mathcal{C}}$ is given $g^\star_{|\mathcal{C}}(x) = \mathbb{1}_{\{\eta(x) \geq 1/2\}}, x \in \mathcal{C}$, that is, the restriction of the Bayes classifier to $\mathcal{C}$. Now it can be easily shown that for any classifier $g \in \mathcal{G}_{\mathcal{C}}$ we have,

$$R(g) - R(g^\star_{|\mathcal{C}}) = \int_{\mathcal{C}} |2\eta(x) - 1| \mathbb{1}_{\{g(x) \neq g^\star_{|\mathcal{C}}(x)\}} \mathrm{d}P_X(x) . \tag{4}$$

Taking expectations on both sides of (4) with $g = \hat{g}_{n,m}$, it follows that

$$\mathbb{E}_{n,m} R(\hat{g}_{n,m}) - R(g^\star_{|\mathcal{C}}) = \mathcal{E}_{\mathcal{C}}(\hat{g}_{n,m}) .$$

Therefore, cluster excess-risk equals the excess-risk of classifiers in $\mathcal{G}_{\mathcal{C}}$. In the sequel, we only consider classifiers $\hat{g}_{n,m} \in \mathcal{G}_{\mathcal{C}}$, that is, classifiers that are defined on $\mathcal{C}$.

We now propose a method to obtain good upper bounds on the cluster excess-risk, taking advantage of the cluster assumption **(CA)**. The idea is to estimate the regions where the sign of $(\eta - 1/2)$ is constant and make a majority vote on each region.

3. Results for Known Clusters

Consider the ideal situation where the family $T_1, T_2, \ldots$, is known and we observe only the labeled sample $\mathbb{X}_l = \{(X_1,Y_1), \ldots, (X_n,Y_n)\}$. Define

$$\mathcal{C} = \bigcup_{j \geq 1} T_j .$$

Under the cluster assumption **(CA)**, the function $x \mapsto \eta(x) - 1/2$ has constant sign on each T_j. Thus a simple and intuitive method for classification is to perform a majority vote on each T_j.

For any $j \geq 1$, define $\delta_j \geq 0, \delta_j \leq 1$ by

$$\delta_j = \int_{T_j} |2\eta(x) - 1| P_X(\mathrm{d}x)\,.$$

We now define our classifier based on the sample $\mathbb{X}_l$. For any $j \geq 1$, define the random variable

$$Z_n^j = \sum_{i=1}^{n} (2Y_i - 1)\, \mathbb{1}_{\{X_i \in T_j\}}\,,$$

and denote by $\hat{g}_n^j$ the function $\hat{g}_n^j(x) = \mathbb{1}_{\{Z_n^j > 0\}}$, for all $x \in T_j$. Consider the classifier defined on $\mathcal{C}$ by

$$\hat{g}_n(x) = \sum_{j \geq 1} \hat{g}_n^j(x)\, \mathbb{1}_{\{x \in T_j\}}\,, \quad x \in \mathcal{C}\,.$$

The following theorem gives rates of convergence for the cluster excess-risk of the classifier $\hat{g}_n$ under **(CA)** that can be exponential in n under a mild additional assumption.

Theorem 3.1 *Let $T_j, j \geq 1$ be a family of measurable sets that satisfy Assumption* **(CA)**. *Then, the classifier $\hat{g}_n$ defined above satisfies*

$$\mathcal{E}_{\mathcal{C}}(\hat{g}_n) \leq 2 \sum_{j \geq 1} \delta_j e^{-n\delta_j^2/2}\,.$$

Moreover, if there exists $\delta > 0$ such that $\delta = \inf_j\{\delta_j : \delta_j > 0\}$, we obtain an exponential rate of convergence:

$$\mathcal{E}_{\mathcal{C}}(\hat{g}_n) \leq 2e^{-n\delta^2/2}\,.$$

In a different framework, Castelli and Cover (1995, 1996) have proved that exponential rates of convergence were attainable for semi-supervised classification. A rapid overview of the proof shows that the rate of convergence $e^{-n\delta^2/2}$ cannot be improved without further assumption. It will be our target in semi-supervised classification. However, we need estimators of the clusters $T_j, j = 1, 2, \ldots$. In the next section we provide the main result on semi-supervised learning, that is when the clusters are unknown but we can estimate them using the unlabeled sample $\mathbb{X}_u$.

4. Main Result

We now deal with a more realistic case where the clusters $T_1, T_2, \ldots,$ are unknown and we have to estimate them using the unlabeled sample $\mathbb{X}_u = \{X_1, \ldots, X_m\}$. We begin by giving a definition of the clusters in terms of density level sets. In this section, we assume that $\mathcal{X}$ has finite Lebesgue measure.

4.1 Definition of the Clusters

Following Hartigan (1975), we propose a definition of clusters that is also compatible with the expression "regions of relatively high P_X" proposed by Seeger (2000).

Assume that P_X admits a density p with respect to the Lebesgue measure on $\mathbb{R}^d$ denoted hereafter by Leb_d. For a fixed $\lambda > 0$, the λ-level set of the density p is defined by

$$\Gamma(\lambda) = \{x \in \mathcal{X} : p(x) \geq \lambda\}\,.$$

On these sets, the density is relatively high. The cluster assumption involves also a notion of connectedness of a set. For any $C \subset \mathcal{X}$, define the binary relation $\mathcal{R}$ on any set C as follows: two points $x, y \in C$ satisfy $x\mathcal{R}y$ if and only if there exists a continuous map $f : [0,1] \to C$, such that $f(0) = x$ and $f(1) = y$. If $x\mathcal{R}y$, we say that x and y are *pathwise connected*. It can be easily show that $\mathcal{R}$ is an equivalence relation and its classes of equivalence are called *connected components* of C. At this point, in view of the formulation of the cluster assumption, it is very tempting to define the clusters as the connected components of C. However, this definition suffers from two major flaws:

1. a connected set cannot be defined up to a set of null Lebesgue measure. Indeed, consider for example the case $d = 1$ and $C = [0,1]$. This set is obviously connected (take the map f equal to the identity on $[0,1]$) but the set $\widetilde{C} = C \setminus \{1/2\}$ is not connected anymore even though C and $\widetilde{C}$ only differ by a set of null Lebesgue measure. In our setup we want to impose connectedness on certain subsets of the λ-level set of the density p which is actually defined up to a set of null Lebesgue measure. Figure 1 (left) is an illustration of a set with one connected component whereas it is desirable to have two clusters.

2. There is no scale consideration in this definition of clusters. When two clusters are too close to each other in a certain sense, we wish identify them as a single cluster. In Figure 1 (right), the displayed set has two connected components whereas we wish to identify only one cluster.

To fix the first flaw, we introduce the following notions. Let $\mathcal{B}(z,r)$ be the d-dimensional closed ball of center $z \in \mathbb{R}^d$ and radius $r > 0$, defined by

$$\mathcal{B}(z,r) = \left\{ x \in \mathbb{R}^d : \|z - x\| \leq r \right\},$$

where $\|\cdot\|$ denotes the Euclidean norm in $\mathbb{R}^d$.

Definition 4.1 *Fix $r_0 \geq 0$ and let $\overline{d}$ be an integer such that $\overline{d} \geq d$. We say that a measurable set $C \subset \mathcal{X}$ is r_0-*standard *if for any $z \in C$ and any $0 \leq r \leq r_0$, we have*

$$\mathrm{Leb}_d\big(\mathcal{B}(z,r) \cap C\big) \geq c_0 r^{\overline{d}}. \tag{5}$$

We now comment upon this definition.

Remark 4.1 *The definition of a standard set has been introduced by Cuevas and Fraiman (1997). This definition ensures that the set C has no "flat" parts which allows to exclude pathological cases such as the one presented on the left hand side of Figure 1.*

Remark 4.2 *The constant c_0 may depend on r_0 and this avoids large-scale shape considerations. Indeed, if the set C is bounded, then for any $z \in C$, $\mathrm{Leb}_d\big(\mathcal{B}(z,r) \cap C\big) = \mathrm{Leb}_d\big(C\big)$ for $r \geq r_0$ where r_0 is the diameter of C. Thus for C to be r_0-standard, we have to impose at least that $c_0 \leq \mathrm{Leb}_d\big(C\big) r_0^{-\overline{d}}$.*

Remark 4.3 *The case $\overline{d} > d$ allows us to include a wide variety of shapes in this definition. Consider the following example where $d = 2$:*

$$C_\delta = \big\{(x,y) : -1 \leq x \leq 1, 0 \leq y \leq |x|^\delta\big\}, \quad \delta > 0.$$

Fix $r \leq \sqrt{2}$ and consider the point $z = (0,0)$. It holds

$$\mathrm{Leb}_d\big(\mathcal{B}(z,r) \cap C_\delta\big) \geq \int_{-r'}^{r'} \min(|x|^\delta, r')\mathrm{d}x\,, \quad \text{where } r' = \frac{r}{\sqrt{2}}\,.$$

For any $|x| \leq r' \leq 1$, we have $|x|^\delta \geq |x|^{(\delta\vee 1)}$ and $|x|^{(\delta\vee 1)} \leq |r'|^{(\delta\vee 1)} \leq r'$. Thus

$$\mathrm{Leb}_d\big(\mathcal{B}(z,r) \cap C_\delta\big) \geq \int_{-r'}^{r'} |x|^{(\delta\vee 1)}\mathrm{d}x = 2(r')^{(\delta\vee 1)+1}\,.$$

We conclude that (5) *is satisfied at $z = (0,0)$ for $\overline{d} = (\delta \vee 1) + 1$. However, notice that*

$$\mathrm{Leb}_d\big(\mathcal{B}(z,r) \cap C_\delta\big) \leq \int_{-r}^{r} |x|^{\delta}\mathrm{d}x = 2r^{(\delta+1)}\,.$$

Thus (5) *is not satisfied at $z = (0,0)$ when $\overline{d} = d$, if $\delta > 1$.*

To overcome the scale problem described in the second flaw, we introduce the notion of s_0-separated sets.

Define the pseudo-distance distance d_∞, between two sets C_1 and C_2 by

$$d_\infty(C_1,C_2) = \inf_{\substack{x\in C_1\\ y\in C_2}} \|x-y\|.$$

We say that two sets C_1, C_2, are *s_0-separated* if $d_\infty(C_1,C_2) > s_0$, for some $s_0 \geq 0$. More generally, we say that the sets $C_1, C_2, \ldots$ are *mutually s_0-separated* if for any $j \neq j'$, C_j and $C_{j'}$ are s_0-separated. On the right hand side of Figure 1, we show an example of two sets that are not s_0-separated for a reasonable s_0. In that particular example, if s_0 is sufficiently small, we would like to identify a single cluster.

We now define s_0-connectedness which is a weaker version of connectedness in the form of a binary relation

Definition 4.2 *Fix $s > 0$ and let $\overset{s}{\underset{C}{\longleftrightarrow}}$ be the binary relation defined on $C \subset \mathcal{X}$ as follows: two points $x, y \in C$ satisfy $x \overset{s}{\underset{C}{\longleftrightarrow}} y$ if and only if there exists a piecewise constant map $f : [0,1] \to C$ such that $f(0) = x$ and $f(1) = y$ and such that f has a* finite *number of jumps that satisfy $\|f(t_+) - f(t_-)\| \leq s$ for any $t \in [0,1]$, where*

$$f(t_+) = \lim_{\substack{\theta\to t\\ \theta>t}} f(\theta) \quad \text{and} \quad f(t_-) = \lim_{\substack{\theta\to t\\ \theta<t}} f(\theta)\,.$$

*If $x \overset{s}{\underset{C}{\longleftrightarrow}} y$, we say that x and y are s-*connected.

Note that x and y are *s-connected* if and only if there exists $z_1, \ldots, z_n \in C$ such that $\|x - z_1\| \leq s$, $\|y - z_n\| \leq s$ and $\|z_i - z_{i+1}\| \leq s$ for any $j = 1, \ldots, n-1$. In other words, there exists a finite sequence of points in C that links x to y and such that two consecutive points in this sequence have distance smaller that s.

Lemma 4.1 *Fix $s > 0$, then the binary relation $\xleftrightarrow[C]{s}$ is an equivalence relation and C can be partitioned into its classes of equivalence. The classes of equivalence of $\xleftrightarrow[C]{s}$ are called s-*connected components of C.

In the next proposition we prove that given a certain scale $s > 0$, it is possible split a r_0-standard and closed set C into a unique partition that is a coarser than the partition defined by the connected components of C and that this partition is finite for such sets.

Proposition 4.1 *Fix $r_0 > 0, s > 0$ and assume that C is a r_0-standard and closed set. Then there exists a unique partition $C_1, \dots C_J$, $J \geq 1$, of C such that*

- *for any $j = 1, \dots, J$ and any $x, y \in C_j$, we have $x \xleftrightarrow[C]{s} y$,*
- *the sets $C_1, \dots, C_J$ are mutually s-separated.*

Remark 4.4 *In what follows we assume that the scale $s = s_0$ is fixed by the statistician. It should be fixed depending on a priori considerations about the scale of the problem. Actually, in the proof of Proposition 4.3, we could even assume that $s_0 = 1/(3 \log m)$, which means that we can have the scale depend on the number of observations. This is consistent with the fact that the finite number of unlabeled observations allows us to have only a blurred vision of the clusters. In this case, we are not able to differentiate between two clusters that are too close to each other but our vision becomes clearer and clearer as m tends to infinity.*

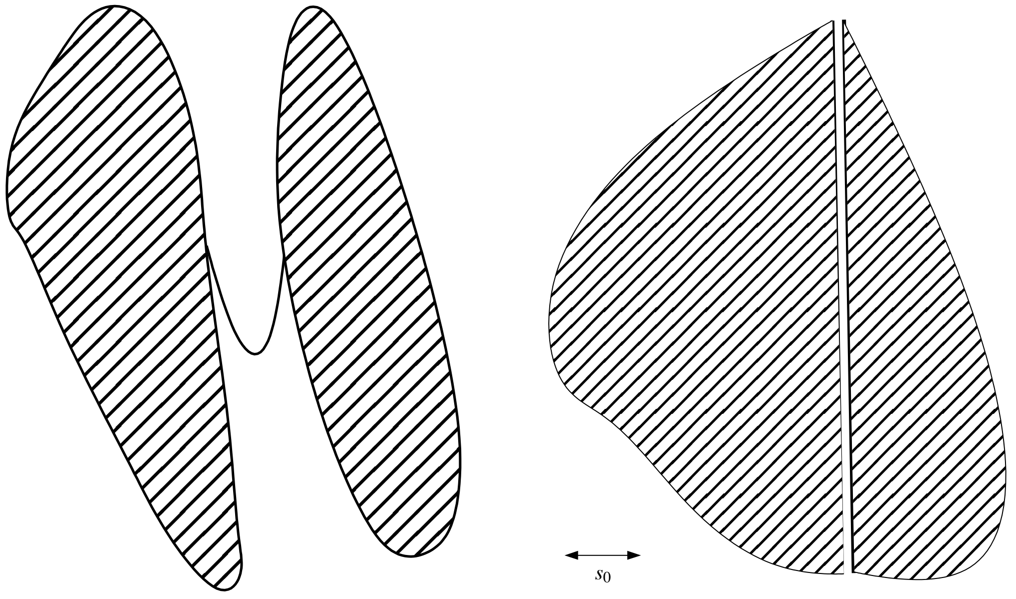


Figure 1: A set that is not r_0-standard for any r_0 (left). A set that has two connected components but only one s_0-connected components (right).

We now formulate the cluster assumption when the clusters are defined in terms of density level sets. In the rest of the section, fix $\lambda > 0$ and let Γ denote the λ-level set of the density p. We also assume in what follows that Γ is closed which is the case if the density p is continuous for example.

Strong Cluster Assumption (SCA) Fix $s_0 > 0$ and $r_0 > 0$ and assume that Γ admits a version that is r_0-standard and closed. Denote by $T_1, \dots, T_J$ the s_0-connected components of this version of Γ. Then the function $x \in \mathcal{X} \mapsto \mathbb{1}_{\{\eta(x) \geq 1/2\}}$ takes a constant value on each of the T_j, $j = 1, \dots J$.

4.2 Estimation of the Clusters

Assume that p is uniformly bounded by a constant $L(p)$ and that $\mathcal{X}$ is bounded. Denote by $\mathbb{P}_m$ and $\mathbb{E}_m$ respectively the probability and the expectation w.r.t the sample $\mathbb{X}_u$ of size m. Assume that we use the sample $\mathbb{X}_u$ to construct an estimator $\hat{G}_m$ of Γ satisfying

$$\mathbb{E}_m\big[\mathrm{Leb}_d(\hat{G}_m \triangle \Gamma)\big] \to 0, \quad m \to +\infty,$$

where $\triangle$ is the sign for the symmetric difference. We call such estimators *consistent* estimators of Γ. Recall that we are interested in identifying the s_0-connected components $T_1, \dots, T_J$ of Γ. That is, we seek a partition of $\hat{G}_m$, denoted here by $\hat{H}_1, \dots, \hat{H}_{J'}$ such that for any $j = 1, \dots, J$, $\hat{H}_j$ is a consistent estimator of T_j and $\mathbb{E}_m\big[\mathrm{Leb}_d(\hat{H}_j)\big] \to 0$ for $j > J$. From Proposition 4.1, we know that for any $1 \leq j, j' \leq J$, $j \neq j'$, we have $d_\infty(T_j, T_{j'}) > s_0$. Let $\bar{s} > s_0$ be defined by

$$\bar{s} = \min_{j \neq j'} d_\infty(T_j, T_{j'})\,. \tag{6}$$

To define the partition $\hat{H}_1, \dots, \hat{H}_{J'}$, it is therefore natural to use a suitable reordering of the $(s_0 + u_m)$-connected components of $\hat{G}_m$, where u_m is a positive sequence that tends to 0 as m tends to infinity. Since the measure of performance $\mathbb{E}_m\big[\mathrm{Leb}_d(\hat{G}_m \triangle \Gamma)\big]$ is defined up to a set of null Lebesgue measure it may be the case that even an estimator $\hat{G}_m$ that satisfies $\mathbb{E}_m\big[\mathrm{Leb}_d(\hat{G}_m \triangle \Gamma)\big] = 0$ has only one $(s_0 + u_m)$-connected components whereas Γ has several s_0-connected components. This happens for example in the case where $\hat{G}_m = \Gamma \cup R$ where R is a set of thin ribbons with null Lebesgue measure that link the s_0-connected components of Γ to each other (see Figure 1, left). If $\hat{G}_m$ were r_0-standard, such configurations would not occur. To have $\hat{G}_m$ more "standard", we apply the following *clipping* transformation: define the set

$$\mathrm{Clip}(\hat{G}_m) = \Big\{x \in \hat{G}_m : \mathrm{Leb}_d\big(\hat{G}_m \cap \mathcal{B}(x, (\log m)^{-1})\big) \leq \frac{(\log m)^{-d}}{m^\alpha}\Big\}.$$

In the sequel, we will only consider the clipped version of $\hat{G}_m$ defined by $\tilde{G}_m = \hat{G}_m \setminus \mathrm{Clip}(\hat{G}_m)$. For any $x \in \tilde{G}_m$, we have

$$\mathrm{Leb}_d\big(\hat{G}_m \cap \mathcal{B}(x, (\log m)^{-1})\big) > \frac{(\log m)^{-d}}{m^\alpha}\,.$$

However, this is not enough to ensure that the union of several s_0-connected components of Γ is not estimated by a single $(s_0 + u_m)$-connected component of $\tilde{G}_m$ due to the magnitude of random fluctuations of $\tilde{G}_m$ around Γ.

To ensure componentwise consistency, we make assumptions on the estimator $\hat{G}_m$. Note that the performance of a density level set estimator $\hat{G}_m$ is measured by the quantity

$$\mathbb{E}_m\big[\mathrm{Leb}_d(\hat{G}_m \triangle \Gamma)\big] = \mathbb{E}_m\big[\mathrm{Leb}_d(\hat{G}_m^c \cap \Gamma)\big] + \mathbb{E}_m\big[\mathrm{Leb}_d(\hat{G}_m \cap \Gamma^c)\big]\,. \tag{7}$$

For some estimators, such as the offset plug-in density level sets estimators presented in Section 5, we can prove that the dominant term in the RHS of (7) is $\mathbb{E}_m\big[\mathrm{Leb}_d(\hat{G}_m^c \cap \Gamma)\big]$. It yields that the probability of having Γ included in the consistent estimator $\hat{G}_m$ is negligible. We now give a precise definition of such estimators.

Definition 4.3 *Let $\hat{G}_m$ be an estimator of Γ and fix $\alpha > 0$. We say that the estimator $\hat{G}_m$ is* consistent from inside *at rate $m^{-\alpha}$ if it satisfies*

$$\mathbb{E}_m\left[\operatorname{Leb}_d(\hat{G}_m \triangle \Gamma)\right] = \widetilde{O}(m^{-\alpha}),$$

and

$$\mathbb{E}_m\left[\operatorname{Leb}_d(\hat{G}_m \cap \Gamma^c)\right] = \widetilde{O}(m^{-2\alpha}).$$

The following proposition ensures that the clipped version of an estimator that is consistent from inside is also consistent from inside at the same rate.

Proposition 4.2 *Fix $\alpha > 0, s_0 > 0$ and let (u_m) be a positive sequence. Assume that $\mathcal{X}$ is bounded and let $\hat{G}_m$ be an estimator of Γ that is consistent from inside at rate $m^{-\alpha}$. Then, the clipped estimator $\tilde{G}_m = \hat{G}_m \setminus \operatorname{Clip}(\hat{G}_m)$ is also consistent from inside a rate $m^{-\alpha}$ and has a finite number $\tilde{K}_m \leq \operatorname{Leb}_d(\mathcal{X})m^{\alpha}$ of (s_0+u_m)-connected components that have Lebesgue measure greater than or equal to $m^{-\alpha}$. Moreover, the (s_0+u_m)-connected components of $\tilde{G}_m$ are mutually $(s_0+\theta u_m)$-separated for any $\theta \in (0,1)$.*

We are now in position to define the estimators of the s_0-connected components of Γ. Define $s_m = s_0 + (3\log m)^{-1}$ and denote by $\tilde{H}_1, \ldots, \tilde{H}_{\tilde{K}_m}$ the s_m-connected components of $\tilde{G}_m$ that have Lebesgue measure greater than or equal to $m^{-\alpha}$. The number $\tilde{K}_m$ depends on $\mathbb{X}_u$ and is therefore random but bounded from above by the deterministic quantity $\operatorname{Leb}_d(\mathcal{X})m^{\alpha}$.

Let $\mathcal{J}$ be a subset of $\{1,\ldots,J\}$. Define $\kappa(j) = \{k = 1,\ldots,\tilde{K}_m : \tilde{H}_k \cap T_j \neq \emptyset\}$ and let $D(\mathcal{J})$ be the event on which the sets $\kappa(j), j \in \mathcal{J}$ are reduced to singletons $\{k(j)\}$ that are disjoint, that is,

$$\begin{aligned} D(\mathcal{J}) &= \Big\{\kappa(j) = \{k(j)\}, k(j) \neq k(j'),\ \forall\, j,j' \in \mathcal{J}, j \neq j'\Big\} \\ &= \Big\{\kappa(j) = \{k(j)\},\ (T_j \cup \tilde{H}_{k(j)}) \cap (T_{j'} \cup \tilde{H}_{k(j')}) = \emptyset,\ \forall\, j,j' \in \mathcal{J}, j \neq j'\Big\}. \end{aligned} \tag{8}$$

In other words, on the event $D(\mathcal{J})$, there is a one-to-one correspondence between the collection $\{T_j\}_{j\in\mathcal{J}}$ and the collection $\big\{\{\tilde{H}_k\}_{k\in\kappa(j)}\big\}_{j\in\mathcal{J}}$. Componentwise convergence of $\tilde{G}_m$ to Γ, is ensured when $D(\{1,\ldots,J\})$ has asymptotically overwhelming probability. The following proposition ensures that $D(\mathcal{J})$ has large enough probability.

Proposition 4.3 *Fix $r_0 > 0$ and $s_0 \geq (3\log m)^{-1}$. Assume that there exists a version of Γ that is r_0-standard and closed. Then, denoting by J the number of s_0-connected components if Γ, for any $\mathcal{J} \subset \{1,\ldots,J\}$, we have*

$$\mathbb{P}_m\big(D^c(\mathcal{J})\big) = \widetilde{O}\big(m^{-\alpha}\big),$$

where $\mathcal{D}(\mathcal{J})$ is defined in (8).

4.3 Labeling the Clusters

From the strong cluster assumption **(SCA)** the clusters are homogeneous regions. To estimate the clusters, we apply the method described above that consists in estimating the s_m-connected components of the clipped estimator $\tilde{G}_m$ and keep only those that have Lebesgue measure greater than or equal to $m^{-\alpha}$. Then we make a majority vote on each homogeneous region. It yields the following procedure.

THREE-STEP PROCEDURE

1. Use the unlabeled data $\mathbb{X}_u$ to construct an estimator $\hat{G}_m$ of Γ that is consistent from inside at rate $m^{-\alpha}$.

2. Define homogeneous regions as the s_m-connected components of $\tilde{G}_m = \hat{G}_m \setminus \mathrm{Clip}(\hat{G}_m)$ (clipping step) that have Lebesgue measure greater than or equal to $m^{-\alpha}$.

3. Assign a single label to each estimated homogeneous region by a majority vote on labeled data.

This method translates into two distinct error terms, one term in m and another term in n. We apply our three-step procedure to build a classifier $\tilde{g}_{n,m}$ based on the pooled sample $(\mathbb{X}_l, \mathbb{X}_u)$. Fix $\alpha > 0$ and let $\hat{G}_m$ be an estimator of the density level set Γ, that is consistent from inside at rate $m^{-\alpha}$. For any $1 \le k \le \tilde{K}_m$, define the random variable

$$Z^k_{n,m} = \sum_{i=1}^{n} (2Y_i - 1)\, \mathbb{1}_{\{X_i \in \tilde{H}_k\}},$$

where $\tilde{H}_k$ is obtained by Step 2 of the three-step procedure. Denote by $\tilde{g}^k_{n,m}$ the function $\tilde{g}^k_{n,m}(x) = \mathbb{1}_{\{Z^k_{n,m} > 0\}}$ for all $x \in \tilde{H}_k$ and consider the classifier defined on $\mathcal{X}$ by

$$\tilde{g}_{n,m}(x) = \sum_{k=1}^{\tilde{K}_m} \tilde{g}^k_{n,m}(x)\, \mathbb{1}_{\{x \in \tilde{H}_k\}}, \quad x \in \mathcal{X}. \tag{9}$$

Note that the classifier $\tilde{g}_{n,m}$ assigns label 0 to any x outside of $\tilde{G}_m$. This is a notational convention and we can assign any value to x on this set since we are only interested in the cluster excess-risk. Nevertheless, it is more appropriate to assign a label referring to a rejection, for example, the values "2"or "R" (or any other value different from $\{0,1\}$). The rejection meaning that this point should be classified using labeled data only. However, when the amount of labeled data is too small, it might be more reasonable not to classify this point at all. This modification is of particular interest in the context of classification with a rejection option when the cost of rejection is smaller than the cost of misclassification (see, for example, Herbei and Wegkamp, 2006). Remark that when there is only a finite number of clusters, there exists $\delta > 0$ such that

$$\delta = \min_{j=1,\dots,J} \left\{\delta_j : \delta_j > 0\right\}. \tag{10}$$

Theorem 4.1 *Fix $\alpha > 0$ and assume that* **(SCA)** *holds. Consider an estimator $\hat{G}_m$ of Γ, based on $\mathbb{X}_u$ that is consistent from inside at rate $m^{-\alpha}$. Then, the classifier $\tilde{g}_{n,m}$ defined in* (9) *satisfies*

$$\mathcal{E}_\Gamma(\tilde{g}_{n,m}) \le \widetilde{O}\left(\frac{m^{-\alpha}}{1-\theta}\right) + \sum_{j=1}^{J} \delta_j e^{-n(\theta\delta_j)^2/2} \le \widetilde{O}\left(\frac{m^{-\alpha}}{1-\theta}\right) + e^{-n(\theta\delta)^2/2}, \tag{11}$$

for any $0 < \theta < 1$ and where $\delta > 0$ is defined in (10).

Note that, since we often have $m \gg n$, the first term in the RHS of (11) can be considered negligible so that we achieve an exponential rate of convergence in n which is almost the same (up to the constant θ in the exponent) as in the case where the clusters are completely known. The constant θ seems to be natural since it balances the two terms.

5. Plug-in Rules for Density Level Sets Estimation

Fix $\lambda > 0$ and recall that our goal is to use the unlabeled sample $\mathbb{X}_u$ of size m to construct an estimator $\hat{G}_m$ of $\Gamma = \Gamma(\lambda) = \{x \in \mathcal{X} : p(x) \geq \lambda\}$, that is consistent from inside at rate $m^{-\alpha}$ for some $\alpha > 0$ that should be as large as possible. A simple and intuitive way to achieve this goal is to use *plug-in estimators* of Γ defined by

$$\hat{\Gamma} = \hat{\Gamma}(\lambda) = \{x \in \mathcal{X} : \hat{p}_m(x) \geq \lambda\} ,$$

where $\hat{p}_m$ is some estimator of p. A straightforward generalization are the *offset plug-in estimators* of $\Gamma(\lambda)$, defined by

$$\tilde{\Gamma}_\ell = \tilde{\Gamma}_\ell(\lambda) = \{x \in \mathcal{X} : \hat{p}_m(x) \geq \lambda + \ell\} ,$$

where $\ell > 0$ is an offset. Clearly, we have $\tilde{\Gamma}_\ell \subset \hat{\Gamma}$. Keeping in mind that we want estimators that are consistent from inside we are going to consider sufficiently large offset $\ell = \ell(m)$.

Plug-in rules is not the only choice for density level set estimation. Direct methods such as empirical excess mass maximization (see, for example, Polonik, 1995; Tsybakov, 1997; Steinwart et al., 2005) are also popular. One advantage of plug-in rules over direct methods is that once we have an estimator $\hat{p}_m$, we can compute the whole collection $\{\tilde{\Gamma}_\ell(\lambda), \lambda > 0\}$, which might be of interest for the user who wants to try several values of λ. Note also that a wide range of density estimators is available in usual software. A density estimator can be parametric, typically based on a mixture model, or nonparametric such as histograms or kernel density estimators. In Section 6, we briefly describe a possible implementation based on existing software that makes use of kernel or nearest neighbors density estimators. To conclude this discussion, remark that the greater flexibility of plug-in rules may result in a poorer learning performance and even though we do not discuss any implementation based on direct methods, it may well be the case that the latter perform better in practice. However, it is not our intent to propose here the best clustering algorithm or the best density level set estimator and we present a simple proof of convergence for offset plug-in rules only for the sake of completeness.

The next assumption has been introduced in Polonik (1995). It is an analog of the margin assumption formulated in Mammen and Tsybakov (1999) and Tsybakov (2004) but for arbitrary level λ in place of $1/2$.

Definition 5.1 *For any* $\lambda, \gamma \geq 0$*, a function* $f : \mathcal{X} \to \mathbb{R}$ *is said to have* γ-exponent at level λ *if there exists a constant* $c^\star > 0$ *such that, for all* $\varepsilon > 0$*,*

$$\text{Leb}_d \{x \in \mathcal{X} : |f(x) - \lambda| \leq \varepsilon\} \leq c^\star \varepsilon^\gamma .$$

When $\gamma > 0$ it ensures that the function f has no flat part at level λ.

The next theorem gives fast rates of convergence for offset plug-in rules when $\hat{p}_m$ satisfies an exponential inequality and p has γ-exponent at level λ. Moreover, it ensures that when the offset ℓ is suitably chosen, the plug-in estimator is consistent from inside.

Theorem 5.1 *Fix* $\lambda > 0, \gamma > 0$ *and* $\Delta > 0$*. Let* $\hat{p}_m$ *be an estimator of the density* p *based on the sample* $\mathbb{X}_u$ *of size* $m \geq 1$ *and let* $\mathcal{P}$ *be a class of densities on* $\mathcal{X}$*. Assume that there exist positive constants* c_1, c_2 *and* $a \leq 1$*, such that for* P_X*-almost all* $x \in \mathcal{X}$*, we have*

$$\sup_{p \in \mathcal{P}} \mathbb{P}_m \left(|\hat{p}_m(x) - p(x)| \geq \delta\right) \leq c_1 e^{-c_2 m^a \delta^2} , \; m^{-a/2} < \delta < \Delta . \qquad (12)$$

Assume further that p has γ-exponent at level λ for any $p \in \mathcal{P}$ and that the offset ℓ is chosen as

$$\ell = \ell(m) = m^{-\frac{a}{2}} \log m \,. \tag{13}$$

Then the plug-in estimator $\tilde{\Gamma}_\ell$ is consistent from inside at rate $m^{-\frac{\gamma a}{2}}$ for any $p \in \mathcal{P}$.

Consider a kernel density estimator $\hat{p}_m^K$ based on the sample $\mathbb{X}_u$ defined by

$$\hat{p}_m^K(x) = \frac{1}{mh^d} \sum_{i=n+1}^{n+m} K\left(\frac{X_i - x}{h}\right), \quad x \in \mathcal{X}\,,$$

where $h > 0$ is the bandwidth parameter and $K : \mathbb{R}^d \to \mathbb{R}$ is a kernel. If p is assumed to have Hölder smoothness parameter $\beta > 0$ and if K and h are suitably chosen, it is a standard exercise to prove inequality of type (12) with $a = 2\beta/(2\beta + d)$. In that case, it can be shown that the rate $m^{-\frac{\gamma a}{2}}$ is optimal in a minimax sense (see Rigollet and Vert, 2006).

6. Discussion

We proposed a formulation of the cluster assumption in probabilistic terms. This formulation relies on Hartigan's (1975) definition of clusters but it can be modified to match other definitions of clusters.

We also proved that there is no hope to improve the classification performance outside of these clusters. Based on these remarks, we defined the cluster excess-risk on which we observe the effect of unlabeled data. Finally we proved that when we have consistent estimators of the clusters, it is possible to achieve exponential rates of convergence for the cluster excess-risk. The theory developed here can be extended to any definition of clusters as long as they can be consistently estimated.

Note that our definition of clusters is parametrized by λ which is left to the user, depending on his trust in the cluster assumption. Indeed, density level sets have the monotonicity property: $\lambda \geq \lambda'$, implies $\Gamma(\lambda) \subset \Gamma(\lambda')$. In terms of the cluster assumption, it means that when λ decreases to 0, the assumption **(SCA)** concerns bigger and bigger sets $\Gamma(\lambda)$ and in that sense, it becomes more and more restrictive. As a result, the parameter λ can be considered as a level of confidence characterizing to which extent the cluster assumption is valid for the distribution P.

The choice of λ can be made by fixing $P_X(\mathcal{C})$, where $\mathcal{C}$ is defined in (1), the probability of the rejection region. We refer to Cuevas et al. (2001) for more details. Note that data-driven choices of λ could be easily derived if we impose a condition on the purity of the clusters, that is, if we are given the δ in (10). Such a choice could be made by decreasing λ until the level of purity is attained. However, any data-driven choice of λ has to be made using the labeled data. It would therefore yield much worse bounds when $n \ll m$.

A possible implementation of the ideas presented in this paper can be designed using existing clustering software such as DBSCAN (Ester et al., 1996) (and its algorithmic improvement called OPTICS Ankerst et al., 1999) or *runt pruning* (Stuetzle, 2003). These three algorithms implement clustering using a definition of clusters that involves density level sets and a certain notion of connectedness. The idea is to use these algorithms on the pooled sample of instances $(X_1, \ldots, X_{n+m}, X)$, where X is the new instance to be classified. As a result every instance will be affected to a cluster by the chosen algorithm. The label for X is then predicted using a majority vote on the labeled instances that are affected to the same cluster as X. Observe that unlike the method described in the paper, the clusters depend on the labeled instances $(X_1,\dots,X_n)$. Proceeding so allows us to use directly existing clustering algorithms without any modification. Since all three algorithms are distance based, we could run them only on unlabeled instance and then affect each labeled instance and the new instance to the same cluster as its nearest neighbor. However, if we assume that $m \gg n$, incorporating labeled instances will not significantly affect the resulting clusters.

We now describe more precisely why these algorithms produce estimated clusters that are related to the s_m-connected components of a plug-in estimator of the density level set. Each algorithm has instances $(X_1,\dots,X_m)$ and several parameters described below as inputs. Note that these clustering algorithms will affect every instance to a cluster. This can be transformed into our framework by removing clusters that contain only one instance.

- DBSCAN has two input parameters: a real number $\varepsilon > 0$ and and integer $M \geq 1$. The basic version of this algorithm proceeds as follows. For a given instance X_i, let $J_\varepsilon(i) \subset \{1,\dots,m\}$ be the set of indexes $j \neq i$ such that $\|X_j - X_i\| \leq \varepsilon$. If $\mathrm{card}(J_\varepsilon(i)) \geq M$ then all instances $X_j, j \in J_\varepsilon(i)$ are affected to the same cluster as X_i and the procedure is repeated with each $X_j, j \in J_\varepsilon(i)$. Otherwise a new cluster is defined and the procedure is repeated with another instance.

  Observe first that the instances X_j that satisfy $\|X_j - X_i\| \leq \varepsilon$ are ε-connected to X_i. Also, define the kernel density estimator $\hat{p}_m$ by:

$$\hat{p}_m(x) = \frac{1}{m\varepsilon^d}\sum_{j=1}^{m} K\Big(\frac{x - X_i}{\varepsilon}\Big),$$

  where $K : \mathbb{R}^d \to \mathbb{R}$ is defined by $K(x) = \mathbb{1}_{\{\|x\|\leq 1\}}$ for any $x \in \mathbb{R}^d$. Then $\mathrm{card}(J_\varepsilon(i)) \geq M$ is equivalent to $\hat{p}_m(X_i) \geq \frac{M+1}{m\varepsilon^d}$. Thus, if we chose $s_0 = \varepsilon - (3\log m)^{-1}$ and $\lambda + \ell(m) = \frac{M+1}{m\varepsilon^d}$, we see that DBSCAN implements our method. Conversely, for given λ and s_0, we can derive the parameters ε and M such that DBSCAN implements our method.

- OPTICS is a modification of DBSCAN that allows the user to compute in an efficient fashion all cluster partitions for different $\varepsilon \leq \varepsilon_0$ for some user specified $\varepsilon_0 > 0$. The user still has to input the chosen value for ε so that from our point of view, the two algorithms are the same.

- Both of the previous algorithms suffer from a major drawback that is inherent to our definition of cluster based on a global level when determining the density level sets. Indeed, in many real data sets, some clusters can only be identified using several density levels. Stuetzle (2003) recently described an algorithm called *runt pruning* that is free from this drawback. Since, it does not implement our method, we do not describe the algorithm in detail but mention it because it implements a more suitable definition of clusters that is also based on connectedness and density level sets. In particular it resolves the problem of choosing λ. It uses a nearest neighbor density estimator as a running horse and uses a single input parameter that corresponds to the scale s_0.

This paper is an attempt to give a proper mathematical framework for the cluster assumption proposed in Seeger (2000). As mentioned above, the definition of clusters that we use here is one among several available and it could be interesting to modify the formulation of the cluster assumption to match other definitions of cluster. In particular, the definition of cluster as s_0-connected components of the λ-level set of the density leaves the problem of choosing λ correctly.

Acknowledgments

The author is most indebted to anonymous referees that contributed to a significant improvement of the paper through their questions and comments.

Appendix A. Proofs

This section contains proofs of the results presented in the paper.

A.1 Proof of Proposition 2.1

Since the distribution of the unlabeled sample $\mathbb{X}_u$ does not depend on η, we have for any marginal distribution P_X,

$$\begin{aligned}
\sup_{\eta\in\Xi}\mathbb{E}_{n,m}\int_{C^c}|2\eta-1|\mathbb{1}_{\{T_{n,m}\neq g^\star\}}\mathrm{d}P_X &= \sup_{\eta\in\Xi}\mathbb{E}_m\mathbb{E}_n\Big[\int_{C^c}|2\eta-1|\mathbb{1}_{\{T_{n,m}\neq g^\star\}}\mathrm{d}P_X\Big|\mathbb{X}_u\Big]\\
&= \mathbb{E}_m\sup_{\eta\in\Xi}\mathbb{E}_n\Big[\int_{C^c}|2\eta-1|\mathbb{1}_{\{T_{n,m}\neq g^\star\}}\mathrm{d}P_X\Big|\mathbb{X}_u\Big]\\
&\geq \inf_{T_n}\sup_{\eta\in\Xi}\mathbb{E}_n\int_{C^c}|2\eta-1|\mathbb{1}_{\{T_n\neq g^\star\}}\mathrm{d}P_X\,,
\end{aligned}$$

where in the last inequality, we used the fact that conditionally on $\mathbb{X}_u$, the classifier $T_{n,m}$ only depends on $\mathbb{X}_l$ and can therefore be written T_n.

A.2 Proof of Theorem 3.1

We can decompose $\mathcal{E}_C(\hat{g}_n)$ into

$$\mathcal{E}_C(\hat{g}_n) = \mathbb{E}_n\sum_{j\geq 1}\int_{T_j}|2\eta(x)-1|\mathbb{1}_{\{\hat{g}_n^j(x)\neq g^\star(x)\}}p(x)\mathrm{d}x\,.$$

Fix $j\in\{1,2,\dots\}$ and assume w.l.o.g. that $\eta\geq 1/2$ on T_j. It yields $g^\star(x)=1,\ \forall x\in T_j$, and since $\hat{g}_n$ is also constant on T_j, we get

$$\begin{aligned}
\int_{T_j}|2\eta(x)-1|\mathbb{1}_{\{\hat{g}_n^j(x)\neq g^\star(x)\}}p(x)\mathrm{d}x &= \mathbb{1}_{\{Z_n^j\leq 0\}}\int_{T_j}(2\eta(x)-1)p(x)\mathrm{d}x\\
&\leq \delta_j\mathbb{1}_{\left\{|\delta_j-\frac{Z_n^j}{n}|\geq\delta_j\right\}}\,.
\end{aligned}\tag{14}$$

Taking expectation $\mathbb{E}_n$ on both sides of (14) we get

$$\begin{aligned}
\mathbb{E}_n\int_{T_j}|2\eta(x)-1|\mathbb{1}_{\{\hat{g}_n^j(x)\neq g^\star(x)\}}p(x)\mathrm{d}x &\leq \delta_j\mathbb{P}_n\Big[|\delta_j-\frac{Z_n^j}{n}|\geq\delta_j\Big]\\
&\leq 2\delta_j e^{-n\delta_j^2/2}\,,
\end{aligned}$$

where we used Hoeffding's inequality to get the last bound. Summing now over j yields the theorem.

A.3 Proof of Lemma 4.1

The binary relation $\xleftrightarrow[C]{s}$ is an equivalence relation if it satisfies reflexivity, symmetry and transitivity.

To prove reflexivity, consider the trivial constant path $f(t)=x$ for all $t\in[0,1]$. We immediately obtain that $x \xleftrightarrow[C]{s} x$.

To prove symmetry, fix $x,y\in C$ such that $x \xleftrightarrow[C]{s} y$ and denote by f_1 the piecewise constant map with n_1 jumps that satisfies $f_1(0)=x$, $f_1(1)=y$ and $\|f_1(t_+)-f_1(t_-)\|\le s$. It is not difficult to see that the map $\tilde{f}_1$ defined by $\tilde{f}_1(t)=f_1(1-t)$ for any $t\in[0,1]$ is piecewise constant with n_1 jumps, satisfies $\tilde{f}_1(0)=y$, $\tilde{f}_1(1)=x$ and $\|\tilde{f}_1(t_+)-\tilde{f}_1(t_-)\|\le s$ for any $t\in[0,1]$, so that $y \xleftrightarrow[C]{s} x$.

To prove transitivity, let $z\in C$ be such that $y \xleftrightarrow[C]{s} z$ and let f_2 be a piecewise constant map with n_2 jumps that satisfies $f_2(0)=y$, $f_2(1)=z$ and $\|f_2(t_+)-f_2(t_-)\|<s$ for any $t\in[0,1]$. Let now $f:[0,1]\to\mathcal{X}$ be the map defined by:

$$f(t)=\begin{cases} f_1(2t) & \text{if } t\in[0,1/2]\\ f_2(2t-1) & \text{if } t\in[1/2,1].\end{cases}$$

This map is obviously piecewise constant with n_1+n_2 jumps and satisfies $f(0)=x, f(1)=z$. Moreover, for any $t\in[0,1]$, f satisfies $\|\tilde{f}(t_+)-\tilde{f}(t_-)\|\le s$.

Thus $\xleftrightarrow[C]{s}$ is an equivalence relation and C can be partitioned into its classes of equivalence.

A.4 Proof of Proposition 4.1

From Lemma 4.1, we know that $\xleftrightarrow[C]{s}$ is an equivalence relation and C can be partitioned into its classes of equivalence denoted by $C_1,C_2,\ldots$. The classes of equivalences $C_1,C_2,\ldots$ obviously satisfy the first point of Proposition 4.1 from the very definition of a class of equivalence.

To check the second point, remark first that since C is a closed set, each C_j, $j\ge1$ is also a closed set. Indeed, fix some $j\ge1$ and let $(x_n, n\ge1)$ be a sequence of points in C_j that converges to $\bar{x}$. Since C is closed, we have $\bar{x}\in C$ so there exists $j'\ge1$ such that $\bar{x}\in C_{j'}$. If $j\ne j'$, then $\|x_n-\bar{x}\|>s$ for any $n\ge1$ which contradicts the fact that x_n converges to $\bar{x}$. Therefore, $\bar{x}\in C_j$ and C_j is closed. Then let C_j and $C_{j'}$, be two classes of equivalence such that $d_\infty(C_j,C_{j'})\le s$. Using the fact that C_j and $C_{j'}$ are closed sets, we conclude that there exist $x\in C_j$ and $x'\in C_{j'}$ such that $\|x-x'\|\le s$ and hence that $x \xleftrightarrow[C]{s} x'$. Thus $C_j=C_{j'}$ and we conclude that for any $C_j,C_{j'}, j\ne j'$, we have $d_\infty(C_j,C_{j'})>s$ and the C_j are mutually s-separated.

We now prove that the decomposition is finite. Since the C_j are mutually s-separated, for any $1\le j\le k$, for any $x_j\in C_j$, the Euclidean balls $\mathcal{B}(x_j,s/3)$ are disjoint. Using the facts that $\mathcal{X}$ is bounded and that C is r_0-standard we obtain,

$$\infty>\mathrm{Leb}_d(\mathcal{X})\ge\sum_{j=1}^{k}\mathrm{Leb}_d\left[\mathcal{B}(x_j,s/3)\cap\mathcal{X}\right]\ge\sum_{j=1}^{k}\mathrm{Leb}_d\left[\mathcal{B}(x_j,s/3)\cap C\right]\ge ck\,,$$

for a positive constant c. Thus we proved the existence of a finite partition

$$C = \bigcup_{j=1}^{J} C_j .$$

It remains to prove that this partition is unique. To this end, we make use of the fundamental theorem of equivalence relations (see, for example, Dummit and Foote, 1991, Prop. 2, page 3) which states that any partition of C corresponds to the classes of equivalences of a unique equivalence relation. Let $\mathcal{P}' = \{C'_1, \dots, C'_{J'}\}$ be a partition of C that satisfies the two points of Proposition 4.1 and denote by $\mathcal{R}'$ the corresponding equivalence relation. We now prove that $\xleftrightarrow[C]{s} \equiv \mathcal{R}'$. From the first point of Proposition 4.1, we easily conclude that if $x\mathcal{R}'y$ then $x \xleftrightarrow[C]{s} y$. Now if we choose $x, y \in C$ such that $x\mathcal{R}'y$ does not hold, then there exist $j \neq j'$ such that $x \in C'_j$ and $y \in C'_{j'}$. If we had $x \xleftrightarrow[C]{s} y$, it would hold $d_\infty(C'_j, C'_{j'}) \leq s$ which contradicts the second point of Proposition 4.1 so $x \xleftrightarrow[C]{s} y$ does not hold. As a consequence we have proved that for any $x, y \in C$, $x\mathcal{R}y$ if and only if $x \xleftrightarrow[C]{s} y$ and the two relations are the same so as their classes of equivalence. This allows us to conclude that $\mathcal{P}' = \mathcal{P}$.

A.5 Proof of Proposition 4.2

Consider a regular grid $\mathcal{G}$ on $\mathbb{R}^d$ with step size $1/\log(m)$ and observe that the Euclidean balls of centers in $\tilde{\mathcal{G}} = \mathcal{G} \cap \mathrm{Clip}(\hat{G}_m)$ and radius $\sqrt{d}/\log(m)$ cover the set $\mathrm{Clip}(\hat{G}_m)$. Since $\mathcal{X}$ is bounded, there exists a constant $c_1 > 0$ such that $\mathrm{card}\{\tilde{\mathcal{G}}\} = c_1(\log m)^d$. Therefore

$$\mathrm{Leb}_d(\mathrm{Clip}(\hat{G}_m)) \leq \sum_{x \in \tilde{\mathcal{G}}} \mathrm{Leb}_d\big(\mathcal{B}(x, \sqrt{d}/\log(m)) \cap \hat{G}_m\big) \leq \frac{c_2(\log m)^{d-\bar{d}}}{m^\alpha},$$

for some positive constant c_2. Therefore, the rate of convergence $\tilde{G}_m$ is the same as that of $\hat{G}_m$. Observe also that $\tilde{G}_m \subset \hat{G}_m$, so that $\tilde{G}_m$ is also consistent from inside.

Assume that $\tilde{G}_m$ can be decomposed in at least a number k of $(s_0 + u_m)$-connected components, $\tilde{H}_1, \dots, \tilde{H}_k$ with Lebesgue measure greater than or equal to $m^{-\alpha}$. It holds

$$\infty > \mathrm{Leb}_d(\mathcal{X}) \geq \sum_{j=1}^{k} \mathrm{Leb}_d(\tilde{T}_j) \geq km^{-\alpha},$$

Therefore, the number of $(s_0 + u_m)$-connected components of $\tilde{G}_m$ with Lebesgue measure greater than or equal to $m^{-\alpha}$ is at most $\mathrm{Leb}_d(\mathcal{X})m^\alpha$.

To prove that the $(s_0 + u_m)$-connected components of $\tilde{G}_m$ are mutually s_0-separated, let $\tilde{T}_1 \neq \tilde{T}_2$ be two $(s_0 + u_m)$-connected components of $\tilde{G}_m$ and fix $x_1 \in \tilde{T}_1, x_2 \in \tilde{T}_2$. We have $\|x_1 - x_2\| > s_0 + u_m$, otherwise $\tilde{T}_1 = \tilde{T}_2$. Thus $d_\infty(\tilde{T}_1, \tilde{T}_2) \geq s_0 + u_m > s_0 + \theta u_m$ for any $u_m > 0, \theta \in (0,1)$. Thus two $(s_0 + u_m)$-connected components of $\tilde{G}_m$ are $(s_0 + \theta u_m)$-separated for any $\theta \in (0,1)$.

A.6 Proof of Proposition 4.3

Define $m_0 = e^{\frac{1}{3(r_0 \wedge s_0)}}$ and denote $D(\mathcal{J})$ by D. Remark that

$$D^c = \bigcup_{j=1}^{J} A_1(j) \cup A_2(j) \cup A_3(j)\,,$$

where

$$\begin{aligned}
A_1(j) &= \{\mathrm{card}[\kappa(j)] = 0\},\\
A_2(j) &= \{\mathrm{card}[\kappa(j)] \geq 2\},\\
A_3(j) &= \bigcup_{j' \neq j} \{\kappa(j) \cap \kappa(j') \neq \emptyset\}\,.
\end{aligned}$$

In words, $A_1(j)$ is the event on which T_j is estimated by none of the $(\tilde{H}_k)_k$, $A_2(j)$ is the event on which T_j is estimated by at least two different elements of the collection $(\tilde{H}_k)_k$ and $A_3(j)$ is the event on which T_j is estimated by an element of the collection $(\tilde{H}_k)_k$ that also estimates another $T_{j'}$ from the collection $(T_j)_j$.

For any $j = 1, \dots, J$, we have

$$A_1(j) = \{\mathrm{card}[\kappa(j)] = 0\} \subset \{T_j \subset \tilde{G}_m \triangle \Gamma\} \subset \{\mathcal{B}(x,r) \cap T_j \subset \tilde{G}_m \triangle \Gamma\}\,,$$

for any $x \in T_j$ and $r > 0$. Remark that from Proposition 4.1, the T_j are mutually s_0-separated so we have $\mathcal{B}(x,r) \cap T_j = \mathcal{B}(x,r) \cap \Gamma$ for any $r \leq s_0$. Thus, for any $m \geq m_0$, it holds $(3\log m)^{-1} \leq s_0 \wedge r_0$ and

$$A_1(j) \subset \{\mathrm{Leb}_d[\mathcal{B}(x,(3\log m)^{-1}) \cap T_j] \leq \mathrm{Leb}_d[\tilde{G}_m \triangle \Gamma]\} \subset \{\mathrm{Leb}_d[\tilde{G}_m \triangle \Gamma] \geq c_0 (3\log m)^{-\bar{d}}\}\,,$$

where in the last inclusion we used the fact that Γ is r_0-standard.

We now treat $A_2(j)$. Assume without loss of generality that $\{1,2\} \subset \kappa(j)$. On $A_2(j)$, there exist $x_1 \in T_j \cap \tilde{H}_1$, $x_n \in T_j \cap \tilde{H}_2$ and a sequence $x_2, \dots, x_{n-1} \in T_j$ such that $\|x_j - x_{j+1}\| \leq s_0$. Observe now that from Proposition 4.2, we have $\|x_1 - x_n\| > s_0 \geq (3\log m)^{-1}$ for $m \geq m_0$. Therefore the integer

$$j^\star = \min\left\{ j \,:\, 2 \leq j \leq n,\, \exists z \in \tilde{H}_1 \text{ s.t. } \|x_j - z\| > (3\log m)^{-1} \right\},$$

is well defined. Moreover, there exists $z_0 \in \tilde{H}_1$ such that $\|x_{j^\star-1} - z_0\| \leq (3\log m)^{-1}$. Now, if there exists $z \in \tilde{H}_k$, for some $k \in \{2, \dots, \tilde{K}_m\}$, such that $\|x_{j^\star} - z\| \leq (3\log m)^{-1}$, then

$$d_\infty(\tilde{H}_1, \tilde{H}_k) \leq \|z_0 - x_{j^\star-1}\| + \|x_{j^\star-1} - x_{j^\star}\| + \|x_{j^\star} - z\| \leq s_0 + 2(3\log m)^{-1}\,.$$

This contradicts the conclusion of Proposition 4.2 which states that $d_\infty(\tilde{H}_1, \tilde{H}_k) > s_0 + \theta(\log m)^{-1}$ for any $k = 2, \dots, \tilde{K}_m$ in particular when $\theta = 2/3$. Therefore we obtain that on $A_2(j)$ there exists $x_{j^\star} \in T_j$ such that

$$\mathcal{B}(x_{j^\star}, (3\log m)^{-1}) \cap \tilde{G}_m = \emptyset\,.$$

It yields

$$\begin{aligned}
A_2(j) &\subset \left\{ \mathcal{B}(x_{j^\star}, (3\log m)^{-1}) \cap T_j \subset \tilde{G}_m \triangle \Gamma \right\}\\
&\subset \left\{ \mathrm{Leb}_d\left[\tilde{G}_m \triangle \Gamma\right] > c_0 (3\log m)^{-\bar{d}} \right\},
\end{aligned}$$

where in the second inclusion used the fact that $\mathcal{B}(x_{j^\star},r)\cap T_j=\mathcal{B}(x_{j^\star},r)\cap\Gamma$ for any $r\le s_0$ and that Γ is r_0-standard.

We now consider the event $A_3(j)$. Assume without loss of generality that $j=1$ and let k be such that $k\in\kappa(1)\cap\kappa(j')$ for some $j'\in\{2,\dots,J\}$. On $A_3(1)$, there exist $y_1\in T_1\cap\tilde H_k$, $y_n\in T_{j'}\cap\tilde H_k$ and a sequence $y_2,\dots,y_{n-1}\in\tilde H_k$ such that $\|y_j-y_{j+1}\|\le s_m$.

Observe now that from Proposition 4.1, we have $\|y_1-y_n\|>s_0\ge(3\log m)^{-1}$ for $m\ge m_0$. Therefore the integer

$$j^\sharp=\min\left\{j\,:\,2\le j\le n,\,\exists z\in T_1\text{ s.t. }\|y_j-z\|>(3\log m)^{-1}\right\},$$

is well defined. Moreover, there exists $z_1\in T_1$ such that $\|y_{j^\sharp-1}-z_1\|\le(3\log m)^{-1}$. Now, if there exists $z\in T_{j'}$ for some $j'\in\{2,\dots,J\}$ such that $\|y_{j^\sharp}-z\|\le(3\log m)^{-1}$, then

$$d_\infty(T_1,T_{j'})\le\|y_{j^\sharp-1}-z_1\|+\|y_{j^\sharp-1}-y_{j^\sharp}\|+(3\log m)^{-1}\le s_0+(\log m)^{-1}<\bar s,$$

for sufficiently large m and where $\bar s$ is defined in (6). This contradicts the definition of $\bar s$ which implies that $d_\infty(T_1,T_{j'})\ge\bar s$ for any $j\in\{2,\dots,J\}$. Therefore we obtain that on $A_3(1)$ there exists $y_{j^\sharp}\in\tilde H_k$ such that $\mathcal{B}(y_{j^\sharp},(3\log m)^{-1})\subset\Gamma^c$. It yields

$$A_3(1)\subset\left\{\text{Leb}_d(\tilde G_m\cap\Gamma^c)\ge\text{Leb}_d(\tilde G_m\cap\mathcal{B}(y_{j^\sharp},(3\log m)^{-1}))\right\}.$$

Since $y_{j^\sharp}\in\tilde G_m\subset\hat G_m$, we have $\text{Leb}_d(\hat G_m\cap\mathcal{B}(y_{j^\sharp},(3\log m)^{-1}))\ge m^{-\alpha}(3\log m)^{-d}$. On the other hand, we have

$$\begin{aligned}\text{Leb}_d(\tilde G_m\cap\mathcal{B}(y_{j^\sharp},(3\log m)^{-1}))&=\text{Leb}_d(\hat G_m\cap\mathcal{B}(y_{j^\sharp},(3\log m)^{-1}))\\&\quad-\text{Leb}_d(\text{Clip}(\hat G_m)\cap\mathcal{B}(y_{j^\sharp},(3\log m)^{-1}))\\&\ge m^{-\alpha}(3\log m)^{-d}-\text{Leb}_d(\hat G_m\cap\Gamma^c)\\&\ge m^{-\alpha}(3\log m)^{-d}-c_3m^{-1.1\alpha}\\&\ge c_4m^{-\alpha}(\log m)^{-d},\end{aligned}$$

where we used the fact that $\hat G_m$ is consistent from inside at rate $m^{-\alpha}$. Hence,

$$A_3(j)=\bigcup_{j'\neq j}\{\kappa(j)\cap\kappa(j')\neq\emptyset\}\subset\left\{\text{Leb}_d(\tilde G_m\cap\Gamma^c)\ge c_5m^{-\alpha}(\log m)^{-d}\right\}.$$

Combining the results for $A_1(j),A_2(j)$ and $A_3(j)$, we have

$$\mathbb{P}_m(D^c)\le\mathbb{P}_m\left\{\text{Leb}_d\left[\tilde G_m\triangle\Gamma\right]>c_0(3\log m)^{-\bar d}\right\}+\mathbb{P}_m\left\{\text{Leb}_d(\tilde G_m\cap\Gamma^c)\ge c_5m^{-\alpha}(\log m)^{-d}\right\}.$$

Using the Markov inequality for both terms we obtain

$$\mathbb{P}_m\left\{\text{Leb}_d\left[\tilde G_m\triangle\Gamma\right]>c_0(3\log m)^{-\bar d}\right\}=\widetilde O\left(m^{-\alpha}\right),$$

and

$$\mathbb{P}_m\left\{\text{Leb}_d(\tilde G_m\cap\Gamma^c)\ge c_5m^{-\alpha}(\log m)^{-d}\right\}=\widetilde O\left(m^{-\alpha}\right),$$

where we used the fact that $\tilde G_m$ is consistent from inside with rate $m^{-\alpha}$. It yields the statement of the proposition.

A.7 Proof of Theorem 4.1

The cluster excess-risk $\mathcal{E}_\Gamma(\tilde{g}_{n,m})$ can be decomposed w.r.t the event D and its complement. It yields

$$\mathcal{E}_\Gamma(\tilde{g}_{n,m}) \le \mathbb{E}_m\left[\mathbb{1}_D\mathbb{E}_n\Big(\int_\Gamma |2\eta(x)-1|\,\mathbb{1}_{\{\tilde{g}_{n,m}(x)\neq g^\star(x)\}}p(x)\mathrm{d}x\Big|\mathbb{X}_u\Big)\right] + \mathbb{P}_m\left(D^c\right) .$$

We now treat the first term of the RHS of the above inequality, that is, on the event D. Fix $j \in \{1,\dots,J\}$ and assume w.l.o.g. that $\eta \ge 1/2$ on T_j. Simply write Z^k for $Z^k_{m,n}$. By definition of D, there is a one-to-one correspondence between the collection $\{T_j\}_j$ and the collection $\{\tilde{H}_k\}_k$. We denote by $\tilde{H}_j$ the unique element of $\{\tilde{H}_k\}_k$ such that $\tilde{H}_j \cap T_j \neq \emptyset$. On D, for any $j = 1,\dots,J$, we have,

$$\begin{aligned}
&\mathbb{E}_n\Big(\int_{T_j} |2\eta(x)-1|\,\mathbb{1}_{\{\tilde{g}^j_{n,m}(x)\neq g^\star(x)\}}p(x)\mathrm{d}x\Big|\mathbb{X}_u\Big)\\
&\qquad\le \int_{T_j\setminus\tilde{G}_m}(2\eta-1)\mathrm{d}P_X + \mathbb{E}_n\Big(\mathbb{1}_{\{Z^j\le 0\}}\int_{T_j\cap\tilde{H}_j}(2\eta-1)\mathrm{d}P_X\Big|\mathbb{X}_u\Big)\\
&\qquad\le L(p)\mathrm{Leb}_d(T_j\setminus\tilde{G}_m) + \delta_j\mathbb{P}_n\big(Z^j\le 0|\mathbb{X}_u\big).
\end{aligned}$$

On the event D, for any $0<\theta<1$, it holds

$$\begin{aligned}
\mathbb{P}_n\big(Z^j\le 0|\mathbb{X}_u\big) &= \mathbb{P}_n\Big(\int_{T_j}(2\eta-1)\mathrm{d}P_X - Z^j \ge \delta_j|\mathbb{X}_u\Big)\\
&\le \mathbb{P}_n\Big(\Big|Z^j - \int_{\tilde{H}_j}(2\eta-1)\mathrm{d}P_X\Big| \ge \theta\delta_j|\mathbb{X}_u\Big)\\
&\quad + \mathbb{1}_{\left\{P_X\left[T_j\triangle\tilde{H}_j\right]\ge(1-\theta)\delta_j\right\}} .
\end{aligned}$$

Using Hoeffding's inequality to control the first term, we get

$$\mathbb{P}_n\big(Z^j\le 0|\mathbb{X}_u\big) \le 2e^{-n(\theta\delta_j)^2/2} + \mathbb{1}_{\left\{P_X\left[T_j\triangle\tilde{H}_j\right]\ge(1-\theta)\delta_j\right\}} .$$

Taking expectations, and summing over j, the cluster excess-risk is upper bounded by

$$\mathcal{E}_\Gamma(\tilde{g}_{n,m}) \le \frac{2L(p)}{1-\theta}\mathbb{E}_m\left[\mathrm{Leb}_d(\Gamma\triangle\tilde{G}_m)\right] + 2\sum_{j=1}^J \delta_j e^{-n(\theta\delta_j)^2/2} + \mathbb{P}_m\left(D^c\right) ,$$

where we used the fact that on D,

$$\sum_{j=1}^J \mathrm{Leb}_d\left[T_j\triangle\tilde{H}_j\right] \le \mathrm{Leb}_d\left[\Gamma\triangle\tilde{G}_m\right] .$$

From Proposition 4.3, we have $\mathbb{P}_m\left(D^c\right) = \widetilde{O}\left(m^{-\alpha}\right)$ and $\mathbb{E}_m\left[\mathrm{Leb}_d(\Gamma\triangle\tilde{G}_m)\right] = \widetilde{O}\left(m^{-\alpha}\right)$ and the theorem is proved.

A.8 Proof of Theorem 5.1

Recall that

$$\tilde{\Gamma}_\ell \triangle \Gamma = \left(\tilde{\Gamma}_\ell \cap \Gamma^c\right) \cup \left(\tilde{\Gamma}_\ell^c \cap \Gamma\right).$$

We begin by the first term. We have

$$\tilde{\Gamma}_\ell \cap \Gamma^c = \left\{x \in \mathcal{X} : \hat{p}_m(x) \geq \lambda + \ell, p(x) < \lambda\right\} \subset \left\{x \in \mathcal{X} : |\hat{p}_m(x) - p(x)| \geq \ell\right\}.$$

The Fubini theorem yields

$$\mathbb{E}_m\left[\mathrm{Leb}_d(\tilde{\Gamma}_\ell \cap \Gamma^c)\right] \leq \mathrm{Leb}_d(\mathcal{X}) \sup_{x \in \mathcal{X}} \mathbb{P}_m\left[|\hat{p}_m(x) - p(x)| \geq \ell\right] \leq c_6 e^{-c_2 m^a \ell^2},$$

where the last inequality is obtained using (12) and $c_6 = c_1 \mathrm{Leb}_d(\mathcal{X}) > 0$. Taking ℓ as in (13) yields for $m \geq \exp(\gamma a / c_2)$,

$$\mathbb{E}_m\left[\mathrm{Leb}_d(\tilde{\Gamma}_\ell \cap \Gamma^c)\right] \leq c_6 m^{-\gamma a}. \tag{15}$$

We now prove that $\mathbb{E}_m\left[\mathrm{Leb}_d(\tilde{\Gamma}_\ell \cap \Gamma^c)\right] = \widetilde{O}\left(m^{-\frac{\gamma a}{2}}\right)$. Consider the following decomposition where we drop the dependence in x for notational convenience,

$$\tilde{\Gamma}_\ell^c \cap \Gamma = B_1 \cup B_2,$$

where

$$B_1 = \left\{\hat{p}_m < \lambda + \ell, p \geq \lambda + 2\ell\right\} \subset \left\{|\hat{p}_m - p| \geq \ell\right\}$$

and

$$B_2 = \left\{\hat{p}_m < \lambda + \ell, \lambda \leq p(x) < \lambda + 2\ell\right\} \subset \left\{|p - \lambda| \leq \ell\right\}.$$

Using (12) and (13) in the same fashion as above we get $\mathbb{E}_m\left[\mathrm{Leb}_d(B_1)\right] = \widetilde{O}\left(m^{-\gamma a}\right)$. The term corresponding to B_2 is controlled using the γ-exponent of density p at level λ. Indeed, we have

$$\mathrm{Leb}_d(B_2) \leq c^\star \ell^\gamma = c^\star (\log m)^\gamma m^{-\frac{\gamma a}{2}} = \widetilde{O}\left(m^{-\frac{\gamma a}{2}}\right).$$

The previous upper bounds for $\mathrm{Leb}_d(B_1)$ and $\mathrm{Leb}_d(B_2)$ together with (15) yield the consistency from inside.

References

M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure. In *Proceedings ACM SIGMOD International Conference on Management of Data*, pages 49–60, 1999.

J.-Y. Audibert and A. Tsybakov. Fast learning rates for plug-in classifiers. *Ann. Statist.*, 34(2), 2007.

M.-F. Balcan and A. Blum. A pac-style model for learning from labeled and unlabeled data. In *Proceedings of the Eighteenth Annual Conference on Learning Theory*, pages 111–126, 2005.

M. Belkin and P. Niyogi. Semi-supervised learning on Riemannian manifolds. *Mach. Learn.*, 56 (1-3):209–239, 2004.

A. Blum and T. M. Mitchell. Combining labeled and unlabeled sata with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pages 92–100, 1998.

S. Boucheron, O. Bousquet, and G. Lugosi. Theory of classification: a survey of some recent advances. *ESAIM Probab. Stat.*, 9:323–375 (electronic), 2005.

O. Bousquet, O. Chapelle, and M. Hein. Measure based regularization. In *Advances in Neural Information Processing Systems*, volume 16, 2004.

V. Castelli and T. M. Cover. On the exponential value of labeled samples. *Pattern Recogn. Lett.*, 16 (1):105–111, 1995.

V. Castelli and T. M. Cover. The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter. *IEEE Trans. Inform. Theory*, 42(6, part 2):2102–2117, 1996.

O. Chapelle, B. Schlkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, Cambridge, Massachusetts, 2006.

O. Chapelle and A. Zien. Semi-supervised classification by low density separation. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 57–64, 2005.

A. Cuevas, M. Febrero, and R. Fraiman. Cluster analysis: a further approach based on density estimation. *Comput. Statist. Data Anal.*, 36(4):441–459, 2001.

A. Cuevas and R. Fraiman. A plug-in approach to support estimation. *Ann. Statist.*, 25(6):2300–2312, 1997.

F. d'Alché Buc, Y. Grandvalet, and C. Ambroise. Semi-supervised marginboost. In *Advances in Neural Information Processing Systems*, volume 14, pages 553–560, 2001.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, New York, 1996.

D. S. Dummit and R. M. Foote. *Abstract Algebra*. Prentice Hall Inc., Englewood Cliffs, New Jersey, 1991.

M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

J. H. Hartigan. *Clustering Algorithms*. John Wiley & Sons, Inc., New York, 1975.

R. Herbei and M. Wegkamp. Classification with rejection option. *Canad. J. Statist.*, 34(4):709–721, 2006.

T. Hertz, A. Bar-Hillel, and D. Weinshall. Boosting margin based distance functions for clustering. In *Proceedings of the Twenty-First International Conference on Machine learning*, 2004.

E. Mammen and A. B. Tsybakov. Smooth discrimination analysis. *Ann. Statist.*, 27(6):1808–1829, 1999.

W. Polonik. Measuring mass concentrations and estimating density contour clusters—an excess mass approach. *Ann. Statist.*, 23(3):855–881, 1995.

M. Rattray. A model-based distance for clustering. In *Proc. of the IEEE-INNS-ENNS Int. Joint Conf. on Neural Networks*, pages 13–16, 2000.

P. Rigollet and R. Vert. Fast rates for plug-in estimators of density level sets. Technical Report 1102, Laboratoire de Probabilités et Modèles Aléatoires de Paris 6, 2006. URL `http://hal.ccsd.cnrs.fr/ccsd-00114180`.

M. Seeger. Learning with labeled and unlabeled data. Technical report, Institute for ANC, Edinburgh, UK, 2000. URL `http://www.dai.ed.ac.uk/~seeger/papers.html.`

I. Steinwart, D. Hush, and C. Scovel. A classification framework for anomaly detection. *J. Mach. Learn. Res.*, 6:211–232, 2005.

W. Stuetzle. Estimating the cluster type of a density by analyzing the minimal spanning tree of a sample. *J. Classification*, 20(1):25–47, 2003.

M. Tipping. Deriving cluster analytic distance functions from Gaussian mixture models. In *Proceedings of the Ninth International Conference on Neural Networks*, pages 815–820, 1999.

A. B. Tsybakov. On nonparametric estimation of density level sets. *Ann. Statist.*, 25(3):948–969, 1997.

A. B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *Ann. Statist.*, 32(1): 135–166, 2004.

S. A. van de Geer. *Applications of Empirical Process Theory*. Cambridge University Press, Cambridge, 2000.

V. N. Vapnik. *Statistical Learning Theory*. Wiley, New-York, 1998.

T. Zhang and F. J. Oles. A probability analysis on the value of unlabeled data for classification problems. In *Proceedings of the Seventeenth International Conference on Machine Learning*, 2000.

X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005. URL `http://www.cs.wisc.edu/~jerryzhu/pub/ssl_survey.pdf`.