Distance Metric Learning VS. Fisher Discriminant Analysis

Babak Alipanahi
David R. Cheriton School of Computer Science
University of Waterloo
200 University Avenue West, Waterloo, Ontario,
Canada N2L 3G1

Michael Biggs and **Ali Ghodsi**
Department of Statistics and Actuarial Science
University of Waterloo
200 University Avenue West, Waterloo, Ontario,
Canada N2L 3G1

Abstract

There has been much recent attention to the problem of learning an appropriate distance metric, using class labels or other side information. Some proposed algorithms are iterative and computationally expensive. In this paper, we show how to solve one of these methods with a closed-form solution, rather than using semidefinite programming. We provide a new problem setup in which the algorithm performs better or as well as some standard methods, but without the computational complexity. Furthermore, we show a strong relationship between these methods and the Fisher Discriminant Analysis.

Introduction

In many fundamental machine learning problems, the Euclidean distances between data points do not represent the desired topology that we are trying to capture. Kernel methods address this problem by mapping the points into new spaces where Euclidean distances may be more useful. An alternative approach is to construct a Mahalanobis distance (quadratic Gaussian metric) over the input space and use it in place of Euclidean distances. This approach can be equivalently interpreted as a linear transformation of the original inputs, followed by Euclidean distance in the projected space. This approach has attracted a lot of recent interest (Xing *et al.* 2003; Bilenko, Basu, & Mooney 2004; Chang & Yeung 2004; Basu, Bilenko, & Mooney 2004; Weinberger, Blitzer, & Saul 2006; Globerson & Roweis 2006; Ghodsi, Wilkinson, & Southey 2007).

In this paper, we introduce a new algorithm which can be solved in closed-form instead of the iterative methods described by Xing *et al.*, Globerson & Roweis and Ghodsi, Wilkinson, & Southey. We also extend the approach by kernelizing it, allowing for non-linear transformations of the metric. We will start by providing a precise definition of the problem before proposing our closed-form solution. Then, we show that our proposed algorithm solves a constraint optimization objective. We also show the effect of this alternative constraint and illustrate the connection between the metric learning problem and Fisher Discernment Analysis (FDA).

Learning Distance Metrics

Problem Definition

The distance metric learning approach has been proposed for both unsupervised and supervised problems. Consider a large data set $\{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ (e.g. a large collection of images) in an unsupervised task. While it would be expensive to have a human examine and label the entire set, it would be practical to select only a small subset of data points and provide information on how they relate to each other. In cases where labeling data is expensive, one may hope that a small investment in pairwise labeling can be extrapolated to the rest of the set.

Note that this information is about the class-equivalence/inequivalence of points but does not necessarily give the actual class labels. Consider a case where there are four points, $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$, and $\mathbf{x}_4$. Given side information that $\mathbf{x}_1$ and $\mathbf{x}_2$ are in the same class, and $\mathbf{x}_3$ and $\mathbf{x}_4$ also share a class, we still cannot be certain whether the four points fall into one or two classes. However, two kinds of class-related side information can be identified. The first is a set of pairs of similar or class-equivalent pairs (i.e. they belong to the same class)

$$S : (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{S} \quad \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are similar}$$

and the second is a set of dissimilar or class-inequivalent pairs (i.e. they belong to different classes)

$$D : (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{D} \quad \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are dissimilar}$$

We then wish to learn a $n \times m$ transformation matrix W ($m \leq n$) which transforms all the points by $f(\mathbf{x}) = W^T\mathbf{x}$. This will induce a Mahalanobis distance d_A over the points

$$d_A(\mathbf{x}_i, \mathbf{x}_j) = \| \mathbf{x}_i - \mathbf{x}_j \|_A = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T A (\mathbf{x}_i - \mathbf{x}_j)} \quad (1)$$

where $A = WW^T$ is a positive semidefinite (PSD) matrix. The distances between points in this new space can then be used with any unsupervised technique (e.g. clustering, embedding).

This setting can be easily extended to the supervised scenario. In this case, the data points with the same label will form the set $\mathcal{S}$, and data points with different labels will construct the set $\mathcal{D}$. The distances between points in this case can then be used with any supervised technique (e.g. classification).

Related works

One proposed method for this problem is described by Xing *et al.* (Xing *et al.* 2003). In this work, a new distance metric is learned by considering side information. Xing *et al.* used side information identifying pairs of points as "similar". They then construct a metric that minimizes the distance between all such pairs of points. At the same time, they attempt to ensure that all "dissimilar" points are separated by some minimal distance. A key observation is that they consider all points not explicitly identified as similar to be dissimilar.

$$\begin{aligned} \min_{A} \quad & \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} \quad \| \mathbf{x}_i - \mathbf{x}_j \|_A^2 \\ \text{s.t.} \quad & \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \quad \| \mathbf{x}_i - \mathbf{x}_j \|_A \geq 1 \\ & A \succeq 0 \end{aligned} \tag{2}$$

Note that using $\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \| \mathbf{x}_i - \mathbf{x}_j \|_A^2 \geq 1$ as a constraint, would always result in a rank one A, and the current constraint (i.e. squared root of Mahalanobis distances) is chosen to avoid that situation. An iterative algorithm for optimizing this objective is presented in which gradient ascent followed by the method of iterative projections is used to satisfy the constraints.

Ghodsi, Wilkinson, & Southey defined the following cost function (Ghodsi, Wilkinson, & Southey 2007), which, attempts to minimize the squared induced distance between similar points, while maximizing the squared induced distance between dissimilar points

$$\begin{aligned} L(A) = \frac{1}{|\mathcal{S}|} \quad & \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} \quad \| \mathbf{x}_i - \mathbf{x}_j \|_A^2 - \\ \frac{1}{|\mathcal{D}|} \quad & \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \quad \| \mathbf{x}_i - \mathbf{x}_j \|_A^2 \end{aligned} \tag{3}$$

The optimization problem then becomes

$$\begin{aligned} \min_{A} \quad & L(A) \\ \text{s.t.} \quad & A \succeq 0 \\ & \text{Tr}(A) = 1 \end{aligned} \tag{4}$$

The first constraint (positive semidefiniteness) ensures a valid metric, and the second constraint excludes the trivial solution where all distances are zero. The cost function is then converted into a linear objective and solved by semidefinite programming (SDP) (Boyd & Vandenberghe 2004).

Note that the constant 1 in the constraint of this method, as well as the constant in the Xing *et al.* method is arbitrary and changing it simply scales the resulting space.

Globerson & Roweis proposed a metric learning method for use in classification tasks (Globerson & Roweis 2006). Similar to the methods proposed in (Xing *et al.* 2003) and (Ghodsi, Wilkinson, & Southey 2007), their approach searches for a metric under which points in the same class are near each other and simultaneously far from points in the other classes. For each training point $\mathbf{x}_i$, a conditional distribution over other points is defined as

$$p^A(j|i) = \frac{e^{(-d_{ij}^A)^2}}{\sum_{k\neq i} e^{(-d_{ik}^A)^2}} \qquad i \neq j$$

where $d_{ij}^A = d_A(\mathbf{x}_i, \mathbf{x}_j)$. In the ideal case, where all points within a class are mapped to a single point and points in other classes are pushed infinitely far away, we would have the ideal "bi-level" distribution

$$p_0(j|i) \propto \begin{cases} 1 & y_i = y_j \\ 0 & y_i \neq y_j \end{cases}$$

where y denotes the label of a training point. The objective is to make the conditional distribution as close as possible to the ideal case. This can be achieved by minimizing the KL divergence between two distributions

$$\min_{A} \sum_i \mathbf{KL}[p_0(j|i)|p^A(j|i)]$$
$$\text{s.t. } A \succeq 0$$

This convex optimization problem is solved by a projected gradient approach similar to the one used in (Xing *et al.* 2003). The algorithm itself is similar to the one used in Neighborhood Component Analysis (NCA) (Goldberger *et al.* 2005), but unlike NCA, the resultant optimization problem is convex.

Analytical solution to metric learning problem

The algorithms proposed in (Xing *et al.* 2003) and (Ghodsi, Wilkinson, & Southey 2007) are both computationally expensive and can not be applied to large or high-dimensional datasets due to this intensive complexity. In this section, we show that the method presented in (Ghodsi, Wilkinson, & Southey 2007) can be solved in closed form and without using SDP. To see this, first replace equation (1) in (3) to obtain

$$\begin{aligned} L(A) \quad = \quad & \tfrac{1}{|\mathcal{S}|} \textstyle\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} (\mathbf{x}_i - \mathbf{x}_j)^T A (\mathbf{x}_i - \mathbf{x}_j) - \\ & \tfrac{1}{|\mathcal{D}|} \textstyle\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} (\mathbf{x}_i - \mathbf{x}_j)^T A (\mathbf{x}_i - \mathbf{x}_j) \end{aligned} \tag{5}$$

Since the terms in the summation of equation (5) are scalar, the objective can be reformulated as

$$\begin{aligned} & (\mathbf{x}_i - \mathbf{x}_j)^T A (\mathbf{x}_i - \mathbf{x}_j) \\ = \quad & \text{Tr}\left((\mathbf{x}_i - \mathbf{x}_j)^T W W^T (\mathbf{x}_i - \mathbf{x}_j)\right) \\ = \quad & \text{Tr}\left(W^T (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T W\right) \end{aligned}$$

This objective should be minimized subject to two constraints (see (4)). We can explicitly solve for W and relax the first constraint. To add the second constraint we make use of the Lagrange multiplier

$$\begin{aligned} \max_{W,\lambda} \quad & \phi(W,\lambda) = \\ & \frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} \text{Tr}\left(W^T (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T W\right) - \\ & \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} \text{Tr}\left(W^T (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T W\right) - \\ & \lambda\left(\text{Tr}(W^T W) - 1\right) \end{aligned} \tag{6}$$

Taking the derivative and setting the result equal to zero implies that

$$(M_S - M_D)\, W = \lambda W$$

where

$$M_S = \frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{S}} (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$$

and

$$M_D = \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{D}} (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$$

This is a standard eigenvector problem and the optimal W is the eigenvector corresponding to the smallest nonzero eigenvalue. Our experimental results also confirm that this closed-form solution is identical to the SDP solution proposed in (Ghodsi, Wilkinson, & Southey 2007). This method always produces rank one solutions, even in the multi-class case. In other words, the original input space will be projected onto a line by this transformation. However, in many cases it is desirable to obtain a compact low-dimensional feature representation of the original input space. Fortunately, this can be achieved easily with a minor modification.

Alternative constraints and relation to FDA

We can require the transformation matrix W to satisfy $W^T W = \mathbf{I}_m$. Similar to the original constraint (i.e. $\mathrm{Tr}(WW^T) = 1$) this constraint does not allow the solution to collapse to the trivial solution. In addition it will avoid a rank one solution and also assures that the different coordinates in the feature space are uncorrelated. Crucially, this optimization can also be done in closed-form. The new optimization problem becomes

$$\min_W \ \mathrm{Tr}\left(W^T (M_S - M_D) W\right) \quad \text{s.t.} \quad W^T W = \mathbf{I}_m \tag{7}$$

If the symmetric and real matrix $(M_S - M_D)$ has eigenvalues $\lambda_1 \leq \ldots \leq \lambda_n$ and eigenvectors $v_1, \ldots, v_n$, then the minimum value of the cost function satisfying the constraint is $\lambda_1 + \ldots + \lambda_m$ and the optimal solution is $W = [v_1, \ldots, v_m]$ (Lütkepohl 1997). It is clear that in this setting, the first direction (the eigenvector corresponding to the smallest nonzero eigenvalue) is always identical to the direction found in the original setting.

The constraints $\mathrm{Tr}(WW^T) = 1$ and $W^T W = \mathbf{I}_m$ are chosen to avoid the trivial zero solution. However, these constraints are fairly arbitrarily. There exist many other constraints that do not allow the solution to collapse to the trivial zero solution. Here we introduce an alternative constraint which make a very close connection between metric learning and Fisher Discriminant Analysis (FDA) (Fisher 1936). Consider the following optimization problem:

$$\min_W \ \mathrm{Tr}\left(W^T (M_S - M_D) W\right) \quad \text{s.t.} \quad W^T M_S W = \mathbf{I}_m \tag{8}$$

Similar to the previous constraints, this constraint prevents the data from collapsing onto a point and removes an arbitrary scaling factor. In addition, the matrix M_S provides a natural measure of covariance and therefore the constraint scales directions of the feature space proportional to their variance. This is in contrast to the previous constraint (see equation (7)) that maps all data points onto the unit hypersphere. Standard methods show that the solution is provided by the matrix of eigenvectors corresponding to the largest eigenvalues of the matrix $M_S^{-1} M_D$.

Since M_S is positive definite, we can decompose it as $M_S = HH^T$. Then, we can rearrange the cost function to be

$$\begin{aligned} &\mathrm{Tr}\left(W^T (M_S - M_D) W\right) \\ &\quad = \mathrm{Tr}\left(W^T (HH^{-1})(M_S - M_D)(H^{T^{-1}} H^T) W\right) \\ &\quad = \mathrm{Tr}\left((W^T H)(\mathbf{I}_n - H^{-1} M_D H^{T^{-1}})(H^T W)\right) \end{aligned}$$

If we take $Q = H^T W$, the optimization problem can be expressed as

$$\min_Q \ \mathrm{Tr}\left(Q^T (\mathbf{I}_n - H^{-1} M_D H^{T^{-1}}) Q\right) \quad \text{s.t.} \quad Q^T Q = \mathbf{I}_m$$

If $(\mathbf{I}_n - H^{-1} M_D H^{T^{-1}})$ which is real and symmetric has eigenvalues $1 - \lambda_n \leq \ldots \leq 1 - \lambda_1$ and orthogonal eigenvectors $v_1, \ldots, v_n$, then the minimum value of the cost function satisfying the constraint is $m - (\lambda_n + \ldots + \lambda_{n-m+1})$ and the optimal solution is $Q = [v_1, \ldots, v_m]$ (Lütkepohl 1997). We can write

$$\begin{aligned} &(H^{-1} M_D H^{T^{-1}}) Q = Q\Lambda, \\ &\Lambda = diag\{\lambda_n, \ldots, \lambda_{n-m+1}\} \end{aligned}$$

If we replace $Q = H^T W$ in the above equation, and multiply on the left by $H^{T^{-1}}$ the result will be

$$\begin{aligned} &(H^{-1} M_D H^{T^{-1}}) H^T W = H^T W \Lambda, \\ &(\underbrace{H^{T^{-1}} H^{-1}}_{M_S^{-1}} M_D) W = W\Lambda \end{aligned} \tag{9}$$

So W is made of the first m eigenvectors of $M_S^{-1} M_D$. It should be noted that eigenvalues of $M_S^{-1} M_D$ and $H^{-1} M_D H^{T^{-1}}$ are the same.

The solution of this optimization problem is closely related to FDA. For a general K-class problem, FDA maps the data into a $(K-1)$-dimensional space such that the distance between projected class means $W^T S_B W$ is maximized while the within class variance $W^T S_W W$ is minimized. Here S_B and S_W are defined as

$$\begin{aligned} S_W &= \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k} (\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)^T \\ S_B &= \sum_{k=1}^{K} N_k (\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T \end{aligned}$$

where $\mathcal{C}_k$ is the set of points in class k and N_k is the cardinality, i.e. the number of data points in class k and $\mathbf{m_k} = \frac{1}{N_k}\sum_{i\in\mathcal{C}_k}\mathbf{x}_i$ and $\mathbf{m} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$. FDA then maximizes an explicit function of the transformation matrix W in the form

$$J(W) = \mathrm{Tr}\left((WS_WW^T)^{-1}(WS_BW^T)\right)$$

The maximum is attained when the matrix W consists of the first m $(m < K-1)$ eigenvectors of $S_W^{-1}S_B$ corresponding to the largest eigenvalues. Interestingly, the solution of distance metric learning method (equation (9)) (i.e. eigenvectors of $M_S^{-1}M_D$) and the solution of FDA (i.e. eigenvectors of $S_W^{-1}S_B$) are closely related. It can be shown that these two methods yield identical results in the binary class problem when both classes have the same number of data points.[1].

Kernelized Metric Learning

In many cases we need to consider non-linear transformations of data in order to apply learning algorithms. One efficient method for doing this is to use a kernel that computes a similarity measure between any two data points. In this section, we show how we can learn a distance metric in the feature space implied by a kernel, allowing our use of side information to be extended to non-linear mappings of the data.

Conceptually, we are mapping the points into a feature space by some non-linear mapping $\phi()$ and then learning a distance metric in that space. Actually performing the mapping is typically undesirable (features may have large or infinite dimensionality), so we employ the well-known *kernel trick,* using some kernel $\mathbf{K}(x_i, x_j)$ that can compute inner products between feature vectors without explicitly constructing them. The squared distances in our objective have the form

$$(\mathbf{x}_i - \mathbf{x}_j)^T WW^T(\mathbf{x}_i - \mathbf{x}_j)$$

This W matrix can be reexpressed as a linear combination of the data points, $W = X\beta$, via the kernel trick. Rewriting our squared distance,

$$\begin{aligned}
&(\mathbf{x}_i - \mathbf{x}_j)^T WW^T(\mathbf{x}_i - \mathbf{x}_j)\\
&\quad= (\mathbf{x}_i - \mathbf{x}_j)^T X\beta\beta^T X^T(\mathbf{x}_i - \mathbf{x}_j)\\
&\quad= (X^T\mathbf{x}_i - X^T\mathbf{x}_j)^T\beta\beta^T(X^T\mathbf{x}_i - X^T\mathbf{x}_j)\\
&\quad= (\mathbf{k}_i - \mathbf{k}_j)^T\beta\beta^T(\mathbf{k}_i - \mathbf{k}_j)
\end{aligned}$$

where $\mathbf{k}_i = X^T x_i$ is the i-th column of $\mathbf{K} = X^T X$. We have now expressed the distance in terms of inner products between data points, which can be computed via the kernel $\mathbf{K}$. Instead of W, we need to optimize β. This will proceed just as in the non-kernelized version. It should be noted that even for low-dimensional but large data sets, $\mathbf{K}$ can be very large. When the solution is not in closed form, applying kernel methods to large problems is not feasible. However, due to the very low computational complexity of the proposed method, we can use kernels on any data set with reasonable size.

[1] In this case $|\mathcal{S}| = \frac{N^2-2N}{4}$ and $|\mathcal{D}| = N^2$. Then if we compute the M_S and M_D and simplify the results, we have $M_S = \frac{2(S_1+S_2)}{N-2}$ and $M_D = \frac{2}{N}(S_1+S_2) + (\mathbf{m}_1-\mathbf{m}_2)(\mathbf{m}_1-\mathbf{m}_2)^T$. Finally $M_S^{-1}M_D$ can be written as $\frac{N-2}{2}(\frac{2}{N}\mathbf{I}_n + (S_1+S_2)^{-1}(\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1-\mathbf{m}_2)^T)$. It is clear that the eigenvectors of $M_S^{-1}M_D$ are the same as the eigenvectors of $S_W^{-1}S_B = (S_1+S_2)^{-1}(\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1-\mathbf{m}_2)^T$

Dataset	# data points	# dimensions	# classes
Wine:	178	13	3
Soybean:	47	35	4
Ion:	351	34	2
Protein:	116	20	6
Balance:	625	5	3
Spam:	461	57	2

Table 1: Description of the UCI datasets used for classification experiments.

Experimental Results

We have investigated the ability for different metric learning algorithms to faithfully incorporate class equivalence information into the learned metric. It is of particular interest to determine if there is much computational penalty for using the iterative methods, or whether the quality of their results make up for the inefficiency.

It has been described how class-equivalence side information can be used to learn a suitable metric for classification. Furthermore, when labeled data is provided, all possible pairings of the input data can be added to the sets of similar and dissimilar pairs. This allows the metric learning algorithms to exploit the same information as any classification algorithm. This is the approach we have taken to compare FDA with the other methods under consideration. We have compared the classification performance of the following algorithms:

- Fisher's Discriminant Analysis (FDA)
- Maximally Collapsing Metric Learning (MCML) (Globerson & Roweis 2006)
- Closed-Form Metric Learning (CFML), which uses the constraint $W^TW = \mathbf{I}$
- CFML-II, which uses the constraint $W^TM_SW = \mathbf{I}$
- The algorithm proposed by Xing *et al.* (Xing *et al.* 2003)

Classification error rates are calculated for six labeled UCI datasets (Asuncion & Newman 2007). The datasets are described in Table 1. The average error rate is computed across 40 random splits of the data; in each split we select a random 70% training set and 30% test set. Every algorithm uses the training set to learn a transformation matrix W which induces a Mahalanobis distance d_A over the input data, where $A = WW^T$. After projecting the data into the transformed space, we use a simple one-nearest-neighbor classifier to propose a label for each test point.

Dataset	Algorithm runtime (seconds)				
	CFML	CFML-II	FDA	MCML	Xing
Wine:	0.05	0.05	0.01	3.2	19.8
Soybean:	0.01	0.01	0.01	3.1	280.4
Ion:	0.7	0.7	0.01	31.5	279.4
Protein:	0.03	0.03	0.01	3.1	17.7
Balance:	0.3	0.3	0.01	16.1	24.4
Spam:	1.6	1.6	0.02	66.8	n/a

Table 2: Average algorithm run-times, in seconds. Note that Xing *et al.*'s algorithm could not run on the Spam dataset due to infeasibility.

Every algorithm that learns such a transformation W can also be used to learn a low-dimensional metric A_m of rank m where $m \leq n$. Thus, A_m can be factorized as $A_m = W_m W_m^T$ where W_m is a transformation to a m-dimensional subspace. Low-dimensional distance metrics are desirable because they can drastically reduce the computational requirements for working with the data, and they can often provide noise reduction as well. To compute an appropriate A_m, the optimal rank-m reconstruction of A can be easily computed from its spectral decomposition: whereas A can be diagonalized as $A = \sum_{i=1}^{n} \lambda_i v_i v_i^T$ we restrict A_m to be $A_m = \sum_{i=1}^{m} \lambda_i v_i v_i^T$ where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$.

The results of our classification experiment are presented in Figure 1 and Figure 2. For each dataset, the average classification error is plotted for several low-dimensional projections of each learned metric.

As noted before, the rank of the FDA solution is K-1 where K is the number of classes in the dataset. Thus the FDA solution will be constant for any m-dimensional projection where $m \geq K - 1$.

Note that in a realistic classification setting, one might select the target dimensionality m using a validation set (subset of the training data) to select the dimensionality which achieves the lowest error rate.

We have run several experiments using the kernel formulation of CFML. However, we have not found a suitable kernel for these datasets which results in much performance improvement.

The average observed running times of these algorithms are shown in Table 2. It is interesting to note that while Xing *et al.*'s method tends to perform poorly, it is also the most computationally intensive. Also, the CFML methods both appear to perform comparably with MCML but they can be computed almost instantaneously, with no iteration necessary.

Conclusions

Many different algorithms have been proposed for learning a distance metric in the presence of side information. This paper has investigated a few algorithms that have proposed complicated cost functions that seemed to necessitate iterative methods. We have proposed a closed-form solution to

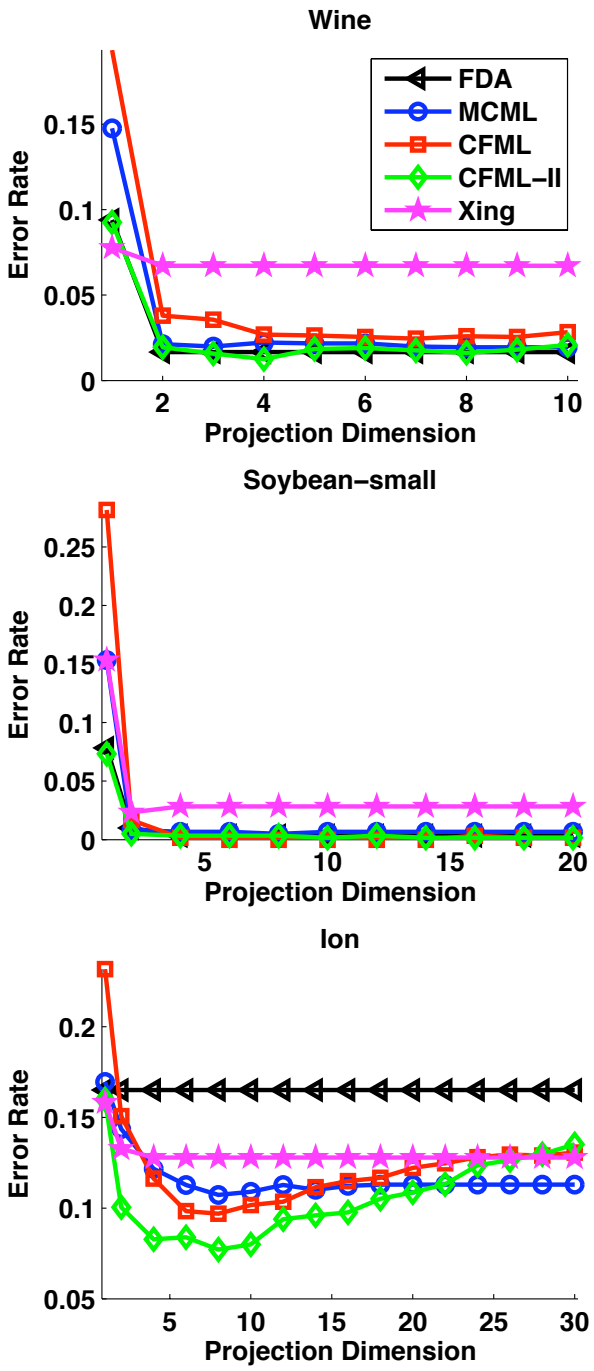

Figure 1: Classification error rate for three of the six UCI datasets. Each learned metric is projected onto a low-dimensional subspace, shown along the x axis.

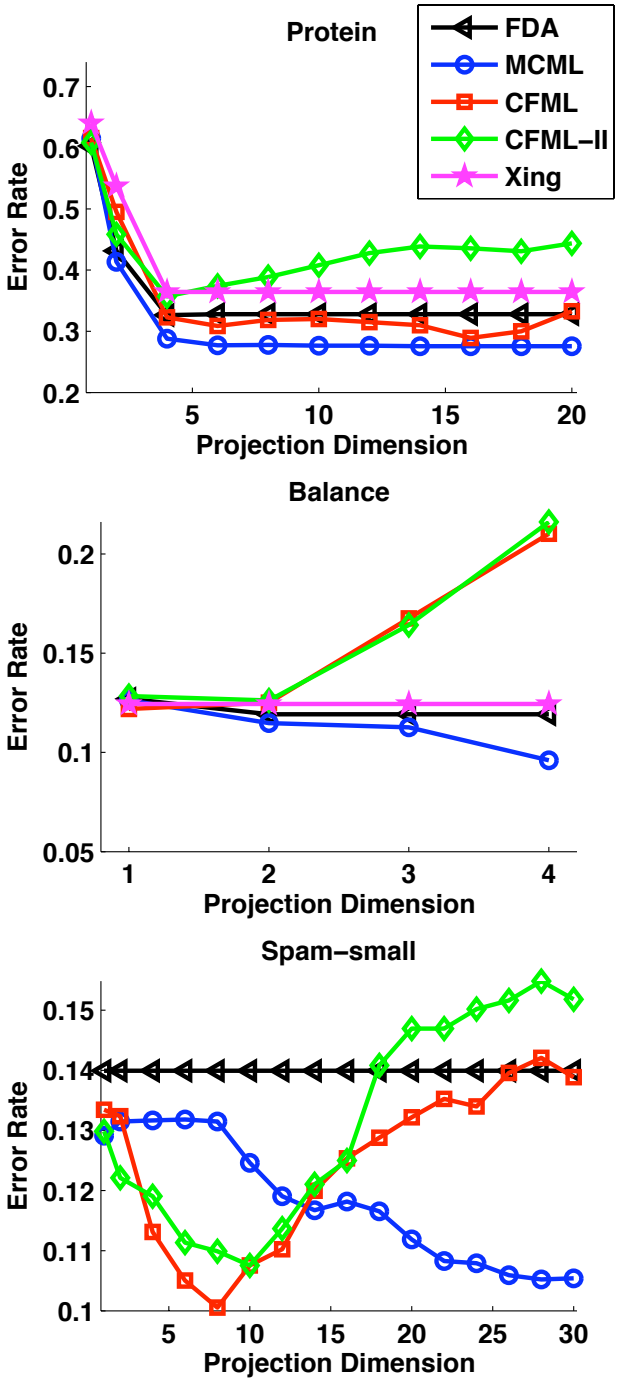

Figure 2: Classification error rate for three of the six UCI datasets. Each learned metric is projected onto a low-dimensional subspace, shown along the x axis.

one algorithm that previously required expensive semidefinite optimization. The new method yields a substantial improvement over Xing's method and FDA. It also has comparable performance to the MCML method but without the runtime inefficiency.

References

Asuncion, A., and Newman, D. 2007. UCI machine learning repository.

Basu, S.; Bilenko, M.; and Mooney, R. J. 2004. A probabilistic framework for semi-supervised clustering. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 59–68. New York, NY, USA: ACM.

Bilenko, M.; Basu, S.; and Mooney, R. J. 2004. Integrating constraints and metric learning in semi-supervised clustering. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, 11. New York, NY, USA: ACM.

Boyd, S., and Vandenberghe, L. 2004. *Convex Optimization*. Cambridge University Press.

Chang, H., and Yeung, D.-Y. 2004. Locally linear metric adaptation for semi-supervised clustering. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, 20. New York, NY, USA: ACM.

Fisher, R. A. 1936. The use of multiple measurements in taxonomic problems. *Annals Eugen.* 7:179–188.

Ghodsi, A.; Wilkinson, D. F.; and Southey, F. 2007. Improving embeddings by flexible exploitation of side information. In Veloso, M. M., ed., *International Joint Conference on Artificial Intelligence*, 810–816.

Globerson, A., and Roweis, S. 2006. Metric learning by collapsing classes. In Weiss, Y.; Schölkopf, B.; and Platt, J., eds., *Advances in Neural Information Processing Systems 18*, 451–458. Cambridge, MA: MIT Press.

Goldberger, J.; Roweis, S.; Hinton, G.; and Salakhutdinov, R. 2005. Neighbourhood components analysis. In Saul, L. K.; Weiss, Y.; and Bottou, L., eds., *Advances in Neural Information Processing Systems 17*, 513–520. Cambridge, MA: MIT Press.

Lütkepohl, H. 1997. *Handbook of Matrices*. New York: Wiley.

Weinberger, K.; Blitzer, J.; and Saul, L. 2006. Distance metric learning for large margin nearest neighbor classification. In Weiss, Y.; Schölkopf, B.; and Platt, J., eds., *Advances in Neural Information Processing Systems 18*, 1473–1480. Cambridge, MA: MIT Press.

Xing, E. P.; Ng, A. Y.; Jordan, M. I.; and Russell, S. 2003. Distance metric learning with application to clustering with side-information. In S. Becker, S. T., and Obermayer, K., eds., *Advances in Neural Information Processing Systems 15*, 505–512. Cambridge, MA: MIT Press.