A Scalable Kernel-Based Algorithm for Semi-Supervised Metric Learning*

Dit-Yan Yeung, Hong Chang, Guang Dai
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
{dyyeung,hongch,daiguang}@cse.ust.hk

Abstract

In recent years, metric learning in the semi-supervised setting has aroused a lot of research interests. One type of semi-supervised metric learning utilizes supervisory information in the form of pairwise similarity or dissimilarity constraints. However, most methods proposed so far are either limited to linear metric learning or unable to scale up well with the data set size. In this paper, we propose a nonlinear metric learning method based on the kernel approach. By applying low-rank approximation to the kernel matrix, our method can handle significantly larger data sets. Moreover, our low-rank approximation scheme can naturally lead to out-of-sample generalization. Experiments performed on both artificial and real-world data show very promising results.

1 Introduction

1.1 Semi-Supervised Learning

In supervised learning, we are given a training sample in the form of input-output pairs. The learning task is to find a functional relationship that maps any input to an output such that disagreement with future input-output observations is minimized. Classification and regression problems are the most common supervised learning problems for discrete-valued and continuous-valued outputs, respectively. In unsupervised learning, we are given a training sample of objects with no output values, with the aim of extracting some structure from them to obtain a concise representation or to gain some understanding of the process that generated the data. Clustering, density estimation, and novelty detection problems are common unsupervised learning problems.

Over the past decade or so, there has been growing interest in exploring new learning problems between the supervised and unsupervised learning extremes. These methods are generally referred to as *semi-supervised learning* methods, although there exist large variations in both the problem formulation and the approach to solve the problem. The semi-supervised learning literature is too large to do a comprehensive review here. Interested readers are referred to some good surveys, e.g., [Zhu, 2006].

One way to categorize many, though not all, semi-supervised learning methods is to consider the type of supervisory information available for learning. Unlike unsupervised learning tasks, supervisory information is available in semi-supervised learning tasks. However, the information is in a form that is weaker than that available in typical supervised learning tasks. One type of (weak) supervisory information assumes that only part (usually a limited part) of the training data are labeled. This scenario is commonly encountered in many real-world applications. One example is the automatic classification of web pages into semantic categories. Since labeling web pages is very labor-intensive and hence costly, unlabeled web pages are far more plentiful on the web. It would be desirable if a classification algorithm can take advantage of the unlabeled data in increasing the classification accuracy. Many *semi-supervised classification* methods [Zhu, 2006] belong to this category.

Another type of supervisory information is even weaker in that it only assumes the existence of some pairwise constraints indicating similarity or dissimilarity relationships between training examples. In video indexing applications, for example, temporal continuity in the data can be naturally used to impose pairwise similarity constraints between successive frames in video sequences. Another example is in proteomic analysis, where protein-protein interactions can naturally be represented as pairwise constraints for the study of proteins encoded by the genes (e.g., *Database of Interacting Proteins* (DIP), http://dip.doe-mbi.ucla.edu/). Yet another example is the anti-spam problem. Recent studies show that more than half of the e-mail in the Internet today is spam, or unsolicited commercial e-mail. One recent approach to spam detection is based on the trustworthiness of social networks [Boykin and Roychowdhury, 2005]. Such social networks can naturally be represented as graphs with edges between nodes representing pairwise relationships in the social networks.

While supervisory information in the form of limited labeled data can be transformed into pairwise similarity and dissimilarity constraints, inverse transformation is in general not possible except for the special case of two-class problems.

*This research has been supported by Competitive Earmarked Research Grant 617404 from the Research Grants Council of the Hong Kong Special Administrative Region, China.

In this sense, the second type of supervisory information is of a weaker form and hence the corresponding learning problem is more difficult to solve. The focus of our paper is on this category of semi-supervised learning problems.

1.2 Semi-Supervised Metric Learning Based on Pairwise Constraints

While many semi-supervised learning methods assume the existence of limited labeled data [Zhu, 2006], there are much fewer methods that can work with pairwise constraints only. We survey the most representative methods in this subsection.

Very often, the pairwise constraints simply state whether two examples belong to the same class or different classes. [Wagstaff and Cardie, 2000] first used such pairwise information for semi-supervised clustering tasks by modifying the standard k-means clustering algorithm to take into account the pairwise similarity and dissimilarity constraints. Extensions have also been made to model-based clustering based on the expectation-maximization (EM) algorithm for Gaussian mixture models [Shental *et al.*, 2004; Lu and Leen, 2005]. However, these methods do not explicitly learn a distance function but seek to satisfy the constraints, typically for clustering tasks. Hence, they are sometimes referred to as constraint-based clustering methods.

As a different category, some methods have been proposed to learn a Mahalanobis metric or some other distance function based on pairwise constraints. [Xing *et al.*, 2003] formulated a convex optimization problem with inequality constraints to learn a Mahalanobis metric and demonstrated performance improvement in a subsequent clustering task. Solving a similar problem to learn a Mahalanobis metric, the relevant component analysis (RCA) algorithm [Bar-Hillel *et al.*, 2003; 2005] was proposed as a simpler and more efficient algorithm than that of [Xing *et al.*, 2003]. However, RCA can make use of similarity constraints only. [Hertz *et al.*, 2004] proposed a distance function learning method called DistBoost. However, there is no guarantee that the distance function learned is a metric. [Bilenko *et al.*, 2004] explored the possibility of integrating constraint-based clustering and semi-supervised clustering based on distance function learning. The distance function learning methods reviewed above either learn a Mahalanobis metric that corresponds to linear transformation only, or learn a distance function that is not a metric. In our previous work [Chang and Yeung, 2004], we proposed a metric learning method that corresponds to nonlinear transformation. While this method is more powerful than the linear methods, the optimization problem is non-convex and hence is more complicated to solve.

In this paper, we focus on the distance function learning approach because distance functions are central to many learning models and algorithms. This also makes it easier to achieve out-of-sample generalization. Moreover, we focus on learning metrics because this allows us to formulate the metric learning problem based on the kernel approach [Schölkopf and Smola, 2002], which provides a disciplined, computationally appealing approach to nonlinear metric learning.

1.3 Organization of Paper

In Section 2, we propose a simple and efficient kernel-based metric learning method based on pairwise similarity constraints. Like many kernel methods, a limitation of this method is that it does not scale up well with the sample size. In Section 3, we address the scalability issue by applying low-rank approximation to the kernel matrix. This extended method can also naturally give rise to out-of-sample generalization, which is addressed in Section 4. We present some experimental results in Section 5 to demonstrate the effectiveness of our metric learning algorithm. Finally, Section 6 concludes the paper.

2 Kernel-Based Metric Learning

2.1 Problem Setup

Let $\{\mathbf{x}_i\}_{i=1}^n$ be a set of n data points in some input space $\mathcal{X}$. Suppose we have a Mercer kernel $\hat{k}$ which induces a nonlinear feature map $\hat{\phi}$ from $\mathcal{X}$ to some reproducing kernel Hilbert space $\mathcal{H}$ [Schölkopf and Smola, 2002]. The corresponding set of feature vectors in $\mathcal{H}$ is $\{\hat{\phi}(\mathbf{x}_i)\}_{i=1}^n$ and the kernel matrix is $\hat{\mathbf{K}} = [\hat{k}(\mathbf{x}_i, \mathbf{x}_j)]_{n\times n} = [\langle\hat{\phi}(\mathbf{x}_i), \hat{\phi}(\mathbf{x}_j)\rangle]_{n\times n}$. Choices for $\hat{k}$ include the Gaussian RBF kernel and the polynomial kernel. We apply a centering transform such that the feature vectors in $\mathcal{H}$ have zero mean. The resulting kernel matrix $\mathbf{K}$ can be computed as $\mathbf{K} = [k(\mathbf{x}_i, \mathbf{x}_j)]_{n\times n} = [\langle\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\rangle]_{n\times n} = \mathbf{H}\hat{\mathbf{K}}\mathbf{H}$, where the centering matrix $\mathbf{H} = \mathbf{I} - (1/n)\mathbf{1}\mathbf{1}^T$ with $\mathbf{I}$ being the $n \times n$ identity matrix and $\mathbf{1}$ being the $n \times 1$ vector of ones.

We consider a type of semi-supervised metric learning in which the supervisory information is given in the form of pairwise similarity constraints.[1] Specifically, we are given a set of point pairs, which is a subset of $\mathcal{X} \times \mathcal{X}$, as $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{x}_j) \mid \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ belong to the same class}\}$. Our goal is to make use of $\mathcal{S}$ to learn a better metric through modifying the kernel so that the performance of some subsequent task (e.g., clustering, classification) based on the metric is improved after kernel learning.

2.2 Kernel Learning

Since the kernel matrix $\mathbf{K}$ is symmetric and positive semi-definite, we can express it as $\mathbf{K} = \sum_{r=1}^p \lambda_r \mathbf{v}_r \mathbf{v}_r^T = \sum_{r=1}^p \lambda_r \mathbf{K}_r$, where $\lambda_1 \geq \cdots \geq \lambda_p > 0$ are the $p \leq n$ positive eigenvalues of $\mathbf{K}$, $\mathbf{v}_1, \ldots, \mathbf{v}_p$ are the corresponding normalized eigenvectors, and $\mathbf{K}_r = \mathbf{v}_r \mathbf{v}_r^T$.

We consider a restricted form of kernel matrix learning by modifying $\mathbf{K}$ through changing the λ_r's while keeping all $\mathbf{K}_r$'s fixed. To ensure that the eigenvalues are nonnegative, we rewrite $\mathbf{K}$ as $\mathbf{K}_{\boldsymbol{\beta}} = [k_{\boldsymbol{\beta}}(\mathbf{x}_i, \mathbf{x}_j)]_{n\times n} = [\langle\phi_{\boldsymbol{\beta}}(\mathbf{x}_i), \phi_{\boldsymbol{\beta}}(\mathbf{x}_j)\rangle]_{n\times n} = \sum_{r=1}^p \beta_r^2 \mathbf{K}_r$, which represents a family of kernel matrices parameterized by $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$.

[1] As we will see later in the formulation of the optimization problem for kernel learning, these constraints are "soft" constraints rather than "hard" constraints in that they are only preferred, not enforced. This makes it easy to handle noisy constraints, i.e., erroneous supervisory information, if we so wish.

We perform kernel learning such that the mean squared Euclidean distance induced by $\mathbf{K}_{\boldsymbol{\beta}}$ between feature vectors in $\mathcal{H}$ corresponding to point pairs in $\mathcal{S}$ is reduced. Thus the criterion function for optimization is

$$\begin{aligned} & J_{\mathcal{S}}(\boldsymbol{\beta}) \\ &= \frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} [(\mathbf{K}_{\boldsymbol{\beta}})_{ii} + (\mathbf{K}_{\boldsymbol{\beta}})_{jj} - 2(\mathbf{K}_{\boldsymbol{\beta}})_{ij}] \\ &= \sum_{r=1}^{p} \beta_r^2 \left[\frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} (\mathbf{b}_i - \mathbf{b}_j)^T \mathbf{K}_r (\mathbf{b}_i - \mathbf{b}_j) \right] \\ &= \boldsymbol{\beta}^T \mathbf{D}_{\mathcal{S}} \boldsymbol{\beta}, \end{aligned} \tag{1}$$

where $\mathbf{b}_i$ is the ith column of the $p \times p$ identity matrix[2] and $\mathbf{D}_{\mathcal{S}}$ is a $p \times p$ diagonal matrix with diagonal entries

$$\begin{aligned} (\mathbf{D}_{\mathcal{S}})_{rr} &= \frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} (\mathbf{b}_i - \mathbf{b}_j)^T \mathbf{K}_r (\mathbf{b}_i - \mathbf{b}_j) \\ &= \frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} [(\mathbf{b}_i - \mathbf{b}_j)^T \mathbf{v}_r]^2 \geq 0. \end{aligned} \tag{2}$$

To prevent $\boldsymbol{\beta}$ from degenerating to the zero vector $\mathbf{0}$ and to eliminate the scaling factor, we minimize the convex function $J_{\mathcal{S}}(\boldsymbol{\beta})$ subject to the linear constraint $\mathbf{1}^T\boldsymbol{\beta} = c$ for some constant $c > 0$. This is a simple convex optimization problem with a quadratic objective function and a linear equality constraint. We introduce a Lagrange multiplier ρ to minimize the following Lagrangian:

$$J_{\mathcal{S}}(\boldsymbol{\beta}, \rho) = J_{\mathcal{S}}(\boldsymbol{\beta}) + \rho(c - \mathbf{1}^T\boldsymbol{\beta}). \tag{3}$$

The optimization problem can be solved easily to give the optimal value of $\boldsymbol{\beta}$ as the following closed-form solution:

$$\boldsymbol{\beta} = \frac{c\mathbf{D}_{\mathcal{S}}^{-1}\mathbf{1}}{\mathbf{1}^T\mathbf{D}_{\mathcal{S}}^{-1}\mathbf{1}}. \tag{4}$$

Note that $\mathbf{D}_{\mathcal{S}}^{-1}$ exists as long as all the diagonal entries of $\mathbf{D}_{\mathcal{S}}$ are positive, which is usually true.[3] We set the constant $c = \sum_{r=1}^{p} \sqrt{\lambda_r}$.

3 Scalable Kernel Learning

The kernel learning method described above requires performing eigendecomposition on $\mathbf{K}$. In case n is very large and hence $\mathbf{K}$ is a large matrix, operations such as eigendecomposition on $\mathbf{K}$ are computationally demanding. In this section, we apply low-rank approximation to extend the kernel learning method so that it scales up well with n.

[2]This indicator variable will be "overloaded" later to refer to a column of any identity matrix whose size is clear from the context.

[3]In case $\mathbf{D}_{\mathcal{S}}$ is really singular (though a rare case), a common way to make it invertible is to add a term $\epsilon\mathbf{I}$ to $\mathbf{D}_{\mathcal{S}}$ where ϵ is a small positive constant.

3.1 Low-Rank Approximation

We apply low-rank approximation to approximate $\mathbf{K}$ by another $n \times n$ symmetric and positive semi-definite matrix $\widetilde{\mathbf{K}}$:

$$\mathbf{K} \simeq \widetilde{\mathbf{K}} = \mathbf{W}\mathbf{L}\mathbf{W}^T, \tag{5}$$

where $\mathbf{W} \in \mathbb{R}^{n\times m}$ is an $n \times m$ matrix and $\mathbf{L} \in \mathbb{R}^{m\times m}$ is an $m \times m$ symmetric and positive semi-definite matrix, with $m \ll n$.

There are different ways to construct $\mathbf{L}$ for low-rank approximation. We consider one way which constructs $\mathbf{L}$ using a subset of the n data points. We refer to these points as *landmarks* [de Silva and Tenenbaum, 2003; Weinberger *et al.*, 2005; Silva *et al.*, 2006]. Without loss of generality, we assume that the n points are ordered in such a way that the first m points form the set of landmarks $\{\mathbf{x}_i\}_{i=1}^m$. We should ensure that all points involved in $\mathcal{S}$ are chosen as landmarks. Other landmarks are randomly sampled from all the data points. Similar to $\mathbf{K}$ in the previous section, $\mathbf{L}$ is obtained here by applying the centering transform to $\hat{\mathbf{L}}$, as $\mathbf{L} = \mathbf{H}\hat{\mathbf{L}}\mathbf{H}$, where $\hat{\mathbf{L}} = [\hat{k}(\mathbf{x}_i, \mathbf{x}_j)]_{m\times m} = [\langle\hat{\phi}(\mathbf{x}_i), \hat{\phi}(\mathbf{x}_j)\rangle]_{m\times m}$ is the upper-left $m \times m$ submatrix of $\hat{\mathbf{K}}$.

We apply eigendecomposition on $\mathbf{L}$ and express it as

$$\mathbf{L} = \sum_{r=1}^{q} \mu_r \boldsymbol{\alpha}_r \boldsymbol{\alpha}_r^T = \mathbf{V}_{\boldsymbol{\alpha}} \mathbf{D}_{\boldsymbol{\mu}} \mathbf{V}_{\boldsymbol{\alpha}}^T, \tag{6}$$

where $\mu_1 \geq \cdots \geq \mu_q > 0$ are the $q \leq m$ positive eigenvalues of $\mathbf{L}$, $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_q$ are the corresponding normalized eigenvectors, $\mathbf{D}_{\boldsymbol{\mu}} = \mathrm{diag}(\mu_1, \ldots, \mu_q)$, and $\mathbf{V}_{\boldsymbol{\alpha}} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_q]$. Substituting (6) into (5), we can rewrite $\widetilde{\mathbf{K}}$ as $\widetilde{\mathbf{K}} = \sum_{r=1}^{q} \mu_r \widetilde{\mathbf{K}}_r$, where $\widetilde{\mathbf{K}}_r = (\mathbf{W}\boldsymbol{\alpha}_r)(\mathbf{W}\boldsymbol{\alpha}_r)^T$.

3.2 Kernel Learning

We apply low-rank approximation to devise a scalable kernel learning algorithm which can be seen as an extension of the algorithm described in Section 2. We use $\widetilde{\mathbf{K}}$ to approximate $\mathbf{K}$ and define the following parameterized family of kernel matrices: $\widetilde{\mathbf{K}}_{\boldsymbol{\beta}} = \sum_{r=1}^{q} \beta_r^2 \widetilde{\mathbf{K}}_r$. Note, however, that $\boldsymbol{\beta}$ is now a $q \times 1$ vector rather than a $p \times 1$ vector.

The optimal value of $\boldsymbol{\beta}$ has the same form as (4), except that the constant c is set to $\sum_{r=1}^{q} \sqrt{\mu_r}$ and $\mathbf{D}_{\mathcal{S}}$ is now a $q \times q$ diagonal matrix with diagonal entries

$$(\mathbf{D}_{\mathcal{S}})_{rr} = \frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} [(\mathbf{b}_i - \mathbf{b}_j)^T (\mathbf{W}\mathbf{v}_r)]^2 \geq 0. \tag{7}$$

3.3 Computing the Embedding Weights

A question that remains to be answered is how to obtain $\mathbf{W}$ for low-rank approximation. We use a method that is similar to locally linear embedding (LLE) [Roweis and Saul, 2000; Saul and Roweis, 2003], with two differences. First, we only use the first part of LLE to obtain the weights for locally linear fitting. Second, we perform locally linear fitting in the kernel-induced feature space $\mathcal{H}$ rather than the input space $\mathcal{X}$.

Let $\mathbf{W} = [w_{ij}]_{n\times m}$ and $\mathbf{w}_i = (w_{i1}, \ldots, w_{im})^T$. If $\mathbf{x}_i$ is a landmark, i.e., $1 \leq i \leq m$, then

$$w_{ij} = \begin{cases} 1 & i = j \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

If $\mathbf{x}_i$ is not a landmark, then we minimize the following function to obtain $\mathbf{w}_i$:

$$E(\mathbf{w}_i) = \left\| \phi(\mathbf{x}_i) - \sum_{\phi(\mathbf{x}_j) \in \mathcal{N}_i} w_{ij} \phi(\mathbf{x}_j) \right\|^2, \quad (9)$$

where $\mathcal{N}_i$ is the set of K nearest landmarks of $\phi(\mathbf{x}_i)$ in $\mathcal{H}$, subject to the constraints $\sum_{\phi(\mathbf{x}_j) \in \mathcal{N}_i} w_{ij} = \mathbf{1}^T \mathbf{w}_i = 1$ and $w_{ij} = 0$ for all $\phi(\mathbf{x}_j) \notin \mathcal{N}_i$. We can rewrite $E(\mathbf{w}_i)$ as

$$\begin{aligned} E(\mathbf{w}_i) &= \sum_{\phi(\mathbf{x}_j), \phi(\mathbf{x}_k) \in \mathcal{N}_i} w_{ij} w_{ik} (\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j))^T \cdot \\ &\qquad (\phi(\mathbf{x}_i) - \phi(\mathbf{x}_k)) \\ &= \mathbf{w}_i^T \mathbf{G}_i \mathbf{w}_i, \end{aligned} \quad (10)$$

where

$$\mathbf{G}_i = [k(\mathbf{x}_i, \mathbf{x}_i) + k(\mathbf{x}_j, \mathbf{x}_k) - k(\mathbf{x}_i, \mathbf{x}_j) - k(\mathbf{x}_i, \mathbf{x}_k)]_{K \times K} \quad (11)$$

is the local Gram matrix of $\mathbf{x}_i$ in $\mathcal{H}$.

To prevent $\mathbf{w}_i$ from degenerating to $\mathbf{0}$, we minimize $E(\mathbf{w}_i)$ subject to the constraints $\sum_{\phi(\mathbf{x}_j) \in \mathcal{N}_i} w_{ij} = \mathbf{1}^T \mathbf{w}_i = 1$ and $w_{ij} = 0$ for all $\phi(\mathbf{x}_j) \notin \mathcal{N}_i$. As above for the kernel learning problem, we solve a convex optimization problem with a quadratic objective function and a linear equality constraint. The Lagrangian with a Lagrange multiplier α is as follows:

$$L(\mathbf{w}_i, \alpha) = \mathbf{w}_i^T \mathbf{G}_i \mathbf{w}_i + \alpha(1 - \mathbf{1}^T \mathbf{w}_i). \quad (12)$$

The closed-form solution for this optimization problem is given by $\mathbf{w}_i = (\mathbf{G}_i^{-1}\mathbf{1})/(\mathbf{1}^T \mathbf{G}_i^{-1} \mathbf{1})$ if $\mathbf{G}_i^{-1}$ exists.[4]

Instead of performing matrix inversion, a more efficient way of finding the solution is to solve the linear system of equations $\mathbf{G}_i \hat{\mathbf{w}}_i = \mathbf{1}$ for $\hat{\mathbf{w}}_i$ and then compute $\mathbf{w}_i$ as $\mathbf{w}_i = \hat{\mathbf{w}}_i / (\mathbf{1}^T \hat{\mathbf{w}}_i)$ to ensure that the equality constraint $\mathbf{1}^T \mathbf{w}_i = 1$ is satisfied.

We assume above that the neighborhood relationships between points in $\mathcal{H}$ and the local Gram matrices remain fixed during the kernel learning process. A simple extension of this basic algorithm is to repeat the above procedure iteratively using the learned kernel at each iteration. Thus the basic algorithm is just a special case of this iterative extension when the number of iterations is equal to one.

4 Out-of-Sample Generalization

The exact form of out-of-sample generalization depends on the operation we want to perform. For example, the n given points are first clustered into $C \geq 2$ classes after kernel learning, and then a new data point $\mathbf{x}$ is classified into one of the C classes. We are interested in the case where both the clustering of the n points and the classification of new points are based on the same Euclidean metric in $\mathcal{H}$.

The key idea of our out-of-sample generalization scheme rests on the observation that kernel principal component analysis (KPCA) [Schölkopf *et al.*, 1998] can be performed on $\{\mathbf{x}_i\}_{i=1}^n$ to obtain an embedding in a q-dimensional subspace $\mathcal{Y}$ of $\mathcal{H}$, so that the non-landmark points $\{\mathbf{x}_i\}_{i=m+1}^n$ and any out-of-sample point $\mathbf{x}$ can be embedded into $\mathcal{Y}$ in the same way.

Let $\{\mathbf{u}_1, \ldots, \mathbf{u}_q\}$ be an orthonormal basis with each $\mathbf{u}_r \in \mathcal{H}$ being a unit vector along the direction of the rth principal component. We define $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_q]$. Then each landmark $\mathbf{x}_i \, (i = 1, \ldots, m)$ can be embedded into $\mathcal{Y}$ to give a q-dimensional vector $\mathbf{y}_i = \mathbf{U}^T \phi(\mathbf{x}_i)$. Since $\mathbf{u}_r = \frac{1}{\sqrt{\mu_r}} \sum_{j=1}^m \alpha_{jr} \phi(\mathbf{x}_j)$, we can express $\mathbf{y}_i$ as $\mathbf{y}_i = \mathbf{D}_{\boldsymbol{\mu}}^{-1/2} \mathbf{V}_{\boldsymbol{\alpha}}^T \mathbf{L} \mathbf{b}_i = \mathbf{D}_{\boldsymbol{\mu}}^{1/2} \mathbf{V}_{\boldsymbol{\alpha}}^T \mathbf{b}_i$. Let $\mathbf{Y}_m = [\mathbf{y}_1, \ldots, \mathbf{y}_m]$. So $\mathbf{Y}_m = \mathbf{D}_{\boldsymbol{\mu}}^{1/2} \mathbf{V}_{\boldsymbol{\alpha}}^T$.

For the non-landmark points $\{\mathbf{x}_i\}_{i=m+1}^n$, from (9), we use $\widetilde{\phi}(\mathbf{x}_i) = \sum_{\phi(\mathbf{x}_j) \in \mathcal{N}_i} w_{ij} \phi(\mathbf{x}_j)$ to approximate $\phi(\mathbf{x}_i)$ and $\widetilde{\mathbf{y}}_i$ to denote the embedding of $\widetilde{\phi}(\mathbf{x}_i)$ in $\mathcal{Y}$. Thus we have

$$\widetilde{\mathbf{y}}_i = \mathbf{U}^T \widetilde{\phi}(\mathbf{x}_i) = \mathbf{Y}_m \mathbf{W}^T \mathbf{b}_i = \mathbf{D}_{\boldsymbol{\mu}}^{1/2} \mathbf{V}_{\boldsymbol{\alpha}}^T \mathbf{W}^T \mathbf{b}_i. \quad (13)$$

Similarly, for any out-of-sample example $\mathbf{x}$, we use $\widetilde{\phi}(\mathbf{x}) = \sum_{\phi(\mathbf{x}_j) \in \mathcal{N}_i} w_j \phi(\mathbf{x}_j)$ to approximate $\phi(\mathbf{x})$ and $\widetilde{\mathbf{y}}$ to denote the embedding of $\widetilde{\phi}(\mathbf{x})$ in $\mathcal{Y}$. The embedding weights $\mathbf{w} = (w_1, \ldots, w_m)^T$ can be determined as in Section 3.3. Similar to the non-landmark points $\{\mathbf{x}_i\}_{i=m+1}^n$, we can obtain

$$\widetilde{\mathbf{y}} = \mathbf{D}_{\boldsymbol{\mu}}^{1/2} \mathbf{V}_{\boldsymbol{\alpha}}^T \mathbf{w}. \quad (14)$$

Based on (13) and (14), the squared Euclidean distance between $\widetilde{\mathbf{y}}_i$ and $\widetilde{\mathbf{y}}$ in $\mathcal{Y}$ before kernel learning can be expressed as

$$\begin{aligned} d^2(\mathbf{x}_i, \mathbf{x}) &= \|\widetilde{\mathbf{y}}_i - \widetilde{\mathbf{y}}\|^2 \\ &= (\mathbf{b}_i^T \mathbf{W} - \mathbf{w}^T) \mathbf{V}_{\boldsymbol{\alpha}} \mathbf{D}_{\boldsymbol{\mu}} \mathbf{V}_{\boldsymbol{\alpha}}^T (\mathbf{W}^T \mathbf{b}_i - \mathbf{w}) \\ &= (\mathbf{b}_i^T \mathbf{W} - \mathbf{w}^T) \mathbf{L} (\mathbf{W}^T \mathbf{b}_i - \mathbf{w}). \end{aligned} \quad (15)$$

The squared Euclidean distance after kernel learning is

$$d_{\boldsymbol{\beta}}^2(\mathbf{x}_i, \mathbf{x}) = (\mathbf{b}_i^T \mathbf{W} - \mathbf{w}^T) \mathbf{V}_{\boldsymbol{\alpha}} \mathbf{D}_{\boldsymbol{\beta}} \mathbf{V}_{\boldsymbol{\alpha}}^T (\mathbf{W}^T \mathbf{b}_i - \mathbf{w}), \quad (16)$$

where $\mathbf{D}_{\boldsymbol{\beta}} = \mathrm{diag}(\beta_1^2, \ldots, \beta_q^2)$.

5 Experimental Results

In this section, we present some experiments we have performed based on both artificial and real-world data.

5.1 Experimental Setup

We compare two versions of our kernel-based metric learning method described in Sections 2 and 3 with RCA [Bar-Hillel *et al.*, 2003; 2005], which is a promising linear metric learning method that usually performs equally well as other computationally more demanding methods. For baseline comparison, we also include two metrics without metric learning. They are the Euclidean metric in the input space and the Euclidean metric in the feature space induced by a Gaussian RBF kernel.

For each data set, we randomly generate 10 different $\mathcal{S}$ sets. For small data sets, we can learn $\mathbf{K}_{\boldsymbol{\beta}}$ without low-rank approximation. For large data sets, in addition to the data points involved in $\mathcal{S}$, we also randomly select some other points as landmarks for learning $\widetilde{\mathbf{K}}_{\boldsymbol{\beta}}$. We use the iterative extension of the scalable kernel learning algorithm with the number of iterations equal to 3. We also measure the change in metric learning performance as the number of landmarks increases.

[4] Similar to $\mathbf{D}_{\mathcal{S}}$ above, we may add $\epsilon \mathbf{I}$ to make sure that $\mathbf{G}_i$ is invertible.

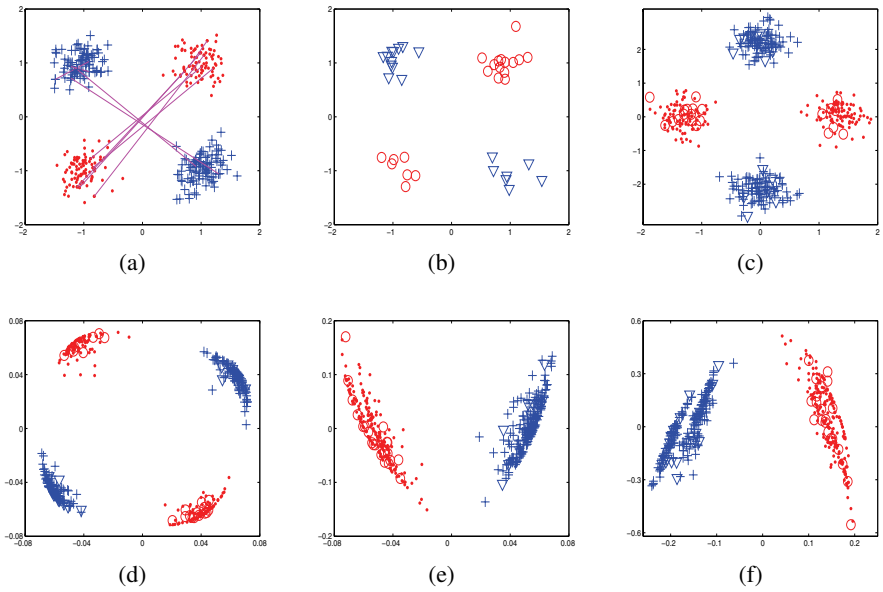

Figure 1: XOR illustration. (a) input data and similarity constraints; (b) new data points; (c) RCA; (d) RBF; (e) $\mathbf{K}_{\boldsymbol{\beta}}$; (f) $\widetilde{\mathbf{K}}_{\boldsymbol{\beta}}$ $(m = 40)$.

5.2 An Illustrative Example

Figure 1 uses the XOR data set to compare the performance of different metrics, some with metric learning. Figure 1(a) shows 360 data points in the input space. The points with the same color and mark belong to the same class. The randomly generated point pairs corresponding to the similarity constraints in $\mathcal{S}$ are shown in solid lines. Figure 1(b) shows 40 new data points in the input space. The results obtained by RCA, RBF kernel and two versions of our method are shown in Figure 1(c)–(f). RCA performs metric learning directly in the input space, which can be generalized to new data using the learned linear transformation. For the kernel methods, we apply KPCA using the (learned) kernel matrix to embed the data points to a 2-dimensional space, as shown in Figure 1(d)–(f). New data points can be embedded using the generalization method described in Section 4. As expected, RCA does not perform satisfactorily for the XOR data set since it can only perform linear transformation. On the other hand, our kernel-based metric learning method can group the points according to their class membership. The result of the scalable kernel learning method with low-rank approximation based on 40 landmarks is almost as good as that of the basic algorithm based on all 400 data points. Moreover, the result on embedding of new data points verifies the effectiveness of our out-of-sample generalization method.

5.3 Quantitative Performance Comparison

Let $\{y_i\}_{i=1}^n$ be the set of true class labels of the n data points. We define the following performance measure:

$$J = \frac{\overline{d}_b}{\overline{d}_w}, \tag{17}$$

where $\overline{d}_b = (1/n_b) \sum_{y_i \neq y_j} \|\mathbf{x}_i - \mathbf{x}_j\|$ is the mean between-class distance with n_b being the number of point pairs with different class labels, and $\overline{d}_w = (1/n_w) \sum_{y_i = y_j} \|\mathbf{x}_i - \mathbf{x}_j\|$ is the mean within-class distance with n_w being the number of point pairs with the same class label. Note that J is closely related to the optimization criterion $J_{\mathcal{S}}$ in (1), except that $J_{\mathcal{S}}$ is defined for the labeled data (i.e., data involved in the pairwise constraints) only while J is for all data assuming the existence of true class labels. For kernel methods, we use $\phi(\mathbf{x}_i)$ or $\widetilde{\phi}(\mathbf{x}_i)$ in place of $\mathbf{x}_i$ and apply the kernel trick to compute the mean distances and hence J. A larger value of J corresponds to a better metric due to its higher class separability.

We first perform some experiments on a much larger XOR data set with 8,000 data points. We randomly select 50 similarity constraints to form $\mathcal{S}$ and measure the metric learning performance in terms of the J value for an increasing number of landmarks. Table 1 shows the results for different metrics. For the metric learning methods (i.e., RCA and our method), we show for each trial the mean (upper) and standard deviation (lower) over 10 random runs corresponding to different $\mathcal{S}$ sets. From the results, we can see that our method outperforms the other methods significantly. Moreover, using more landmarks generally gives better results.

We further perform some experiments on real-world data sets. One of them is the Isolet data set from the UCI Machine Learning Repository which contains 7,797 isolated spoken English letters belonging to 26 classes, with each letter represented as a 617-dimensional vector. Other data sets are handwritten digits from the MNIST database.[5] The digits in the database have been size-normalized and centered to 28×28

[5] http://yann.lecun.com/exdb/mnist/

Table 1: Performance comparison in terms of J value for XOR data set ($|\mathcal{S}| = 50$, $m = 100{:}100{:}800$).

INPUT DATA	RBF	RCA	OUR METHOD							
			$m = 100$	$m = 200$	$m = 300$	$m = 400$	$m = 500$	$m = 600$	$m = 700$	$m = 800$
1.2460	1.4253	1.2552	2.8657	3.0886	3.5640	3.8422	4.2378	4.6395	4.8334	4.7463
		± 0.19	± 1.28	± 0.42	± 0.78	± 1.25	± 0.99	± 0.68	± 0.90	± 0.65

Table 2: Performance comparison in terms of J value for Isolet and MNIST data sets ($|\mathcal{S}| = 50$, $m = 100$).

	ISOLET	MNIST								
		$\{0, 1\}$	$\{1, 3\}$	$\{1, 5\}$	$\{1, 7\}$	$\{1, 9\}$	$\{0, 1, 2\}$	$\{6, 7, 8\}$	$\{0, 1, 9\}$	$\{3, 4, 5, 6\}$
INPUT DATA	1.3888	1.3914	1.2124	1.1920	1.2458	1.2379	1.2408	1.1534	1.2779	1.1162
RBF	1.2477	1.2460	1.1447	1.1357	1.1570	1.1548	1.1598	1.0963	1.1796	1.0729
RCA	1.3049	1.2970	1.1779	1.1705	1.1820	1.2086	1.1844	1.1207	1.2087	1.0793
	± 0.0030	± 0.0324	± 0.0270	± 0.0283	± 0.0399	± 0.0391	± 0.0311	± 0.0160	± 0.0200	± 0.0110
OUR METHOD	2.7938	2.9078	1.7070	1.4015	1.5463	1.7023	1.8620	1.6233	1.9608	1.2945
	± 0.0430	± 0.4614	± 0.2252	± 0.1400	± 0.2055	± 0.3567	± 0.3160	± 0.1996	± 0.1884	± 0.0667

gray-level images. Hence the dimensionality of the input space is 784. In our experiments, we randomly choose 2,000 images for each digit from a total of 60,000 digit images in the MNIST training set. We use 50 similarity constraints and 100 landmarks in the experiments. The results for Isolet and different digit data sets are shown in Table 2. From the results, we can again see that the metric learned by our method is the best in terms of the J measure.

6 Concluding Remarks

We have presented a simple and efficient kernel-based semi-supervised metric learning method based on supervisory information in the form of pairwise similarity constraints. Not only does it scale up well with the data set size, it can also naturally lead to out-of-sample generalization. Although previous studies by other researchers showed that pairwise dissimilarity constraints usually cannot help much in many real-world applications, there are situations when incorporating them may still be helpful and hence we plan to extend our method to incorporate dissimilarity constraints as well. In our low-rank approximation scheme, besides including those points involved in $\mathcal{S}$, we also randomly sample some other points as landmarks. A recent study [Silva *et al.*, 2006] shows that non-uniform sampling of landmarks for manifold learning can give parsimonious approximations using only very few landmarks. We will pursue research along this direction in our future work.

References

[Bar-Hillel *et al.*, 2003] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 11–18, Washington, DC, USA, 21–24 August 2003.

[Bar-Hillel *et al.*, 2005] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning a Mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, 6:937–965, 2005.

[Bilenko *et al.*, 2004] M. Bilenko, S. Basu, and R.J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the Twenty-First International Conference on Machine Learning*, pages 81–88, Banff, Alberta, Canada, 4–8 July 2004.

[Boykin and Roychowdhury, 2005] P.O. Boykin and V.P. Roychowdhury. Leveraging social networks to fight spam. *IEEE Computer*, 38(4):61–68, 2005.

[Chang and Yeung, 2004] H. Chang and D.Y. Yeung. Locally linear metric adaptation for semi-supervised clustering. In *Proceedings of the Twenty-First International Conference on Machine Learning*, pages 153–160, Banff, Alberta, Canada, 4–8 July 2004.

[de Silva and Tenenbaum, 2003] V. de Silva and J.B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 705–712. MIT Press, Cambridge, MA, USA, 2003.

[Hertz *et al.*, 2004] 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*, pages 393–400, Banff, Alberta, Canada, 4–8 July 2004.

[Lu and Leen, 2005] Z. Lu and T.K. Leen. Semi-supervised learning with penalized probabilistic clustering. In L.K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 849–856. MIT Press, Cambridge, MA, USA, 2005.

[Roweis and Saul, 2000] S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[Saul and Roweis, 2003] L.K. Saul and S.T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

[Schölkopf and Smola, 2002] B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, USA, 2002.

[Schölkopf *et al.*, 1998] B. Schölkopf, A. Smola, and K.R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[Shental *et al.*, 2004] N. Shental, A. Bar-Hillel, T. Hertz, and D. Weinshall. Computing Gaussian mixture models with EM using equivalence constraints. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, USA, 2004.

[Silva *et al.*, 2006] J.G. Silva, J.S. Marques, and J.M. Lemos. Selecting landmark points for sparse manifold learning. In *Advances in Neural Information Processing Systems 18*. MIT Press, Cambridge, MA, USA, 2006. To appear.

[Wagstaff and Cardie, 2000] K. Wagstaff and C. Cardie. Clustering with instance-level constraints. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1103–1110, Stanford, CA, USA, 29 June – 2 July 2000.

[Weinberger *et al.*, 2005] K.Q. Weinberger, B.D. Packer, and L.K. Saul. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 381–388, Barbados, 6–8 January 2005.

[Xing *et al.*, 2003] E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 505–512. MIT Press, Cambridge, MA, USA, 2003.

[Zhu, 2006] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, University of Wisconsin – Madison, Department of Computer Science, Madison, Wisconsin, USA, September 7 (last modified) 2006.