Low Rank Metric Learning with Manifold Regularization

Guoqiang Zhong, Kaizhu Huang, Cheng-Lin Liu
National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences
Beijing 100190, P. R. China
{gqzhong, kzhuang, liucl}@nlpr.ia.ac.cn

***Abstract*—In this paper, we present a semi-supervised method to learn a low rank Mahalanobis distance function. Based on an approximation to the projection distance from a manifold, we propose a novel *parametric* manifold regularizer. In contrast to previous approaches that usually exploit side information only, our proposed method can further take advantages of the intrinsic manifold information from data. In addition, we focus on learning a metric of low rank directly; this is different from traditional approaches that often enforce the l_1 norm on the metric. The resulting configuration is convex with respect to the manifold structure and the distance function, respectively. We solve it with an alternating optimization algorithm, which proves effective to find a satisfactory solution. For efficient implementation, we even present a fast algorithm, in which the manifold structure and the distance function are learned independently without alternating minimization. Experimental results over 12 standard UCI data sets demonstrate the advantages of our method.**

***Keywords*-Semi-supervised metric learning; manifold regularization; low rank; linearly constrained nuclear norm minimization; fixed-point iterative algorithm**

I. Introduction

Distance metric is an essential component in many machine learning algorithms, such as nearest neighbor classifiers and K-means. In order to deliver satisfactory results, selecting an appropriate distance metric often plays a critical role. To this end, many excellent methods for distance metric learning have been proposed [1]–[8]. From the perspective of specific learning paradigms, these methods can be classified into three categories: supervised metric learning, unsupervised metric learning and semi-supervised metric learning. Supervised metric learning methods are often utilized to improve the performance of nearest neighbor classifiers, such that data points from the same class are encouraged to be close while those from different classes to be far apart [1], [2], [5], [9], [10]. Unsupervised metric learning methods are in general developed to exploit the underlying manifold structure of data, so as to address unsupervised problems, such as clustering [11] or visualization [12], [13]. Semi-supervised metric learning combines the idea of supervised metric learning and unsupervised metric learning, which integrates both label information from labeled data and geometric information from unlabeled data for classification, clustering or retrieval problems [6], [14]. The merit of semi-supervised metric learning stems from that it utilizes as much as possible geometric information of the data to alleviate the problem arose from the label information deficiency.

Whilst existing methods for distance metric learning have been shown to perform well across various learning paradigms, most of them exploit merely side information from data (e.g., similar or dissimilar constraints and genuine labels) [2], [5], [10]. Recently, [6] and [14] proposed approaches that can utilize the geometric information via the Laplacian regularizer. However, these approaches either formulated the problem as a semi-definite programming (SDP) problem or learned the metric heuristically. SDP is computationally difficult to optimize, while the heuristic method may not guarantee the global solution. In addition, previous methods usually learn a metric without enforcing the low rank (or simply regularizing with the l_1-norm), which however proves effective to suppress noise and to some extent reflect intrinsic structure of the data.

In this paper, we propose a novel semi-supervised algorithm to learn a low rank Mahalanobis distance function directly. Based on an approximation to the projection distance from a manifold, we propose a parametric manifold regularizer to the metric learning model. In contrast to previous approaches that usually exploit side information only, our proposed method can further take advantages of the intrinsic manifold information from data. This significantly improves the performance of learned metric. Moreover, we adopt the *linearly constrained nuclear norm minimization* which directly learns a low rank metric. This is distinctive with those standard methods (either enforcing l_1-norm regularization or even no low rank constraints). The resulting formulation is convex with respect to (w.r.t.) the manifold structure and the distance function, respectively. It can be solved using an alternating optimization algorithm, which proves effective to find a satisfactory solution. More precisely, when the manifold structure is fixed, the problem can be solved with a fixed-point iterative algorithm that is guaranteed to converge globally. When the distance metric is fixed, this problem is a standard quadratic programming problem, and can be solved with classic algorithms or online softwares.

In addition, we find a tight upper bound on the objective function to learn the manifold structure, that directly leads to a new *nonparametric* manifold regularizer to our model.

Substituting this nonparametric manifold regularizer for the original parametric one, we can derive a fast algorithm for our original model. In this fast algorithm, the manifold structure is learned directly from data, while the distance metric can be learned with a fixed-point iterative method based on the learned manifold structure. Since this algorithm avoids the alternating minimization procedures, it is fairly fast compared to the original one.

The rest of this paper is organized as follows. In Section II, we introduce notation and formulation of our model in detail. The optimization algorithm is presented in Section III, while a fast solver of our model is described in Section IV. Section V shows the experimental settings and results. Finally, we conclude this paper in Section VI.

II. Problem Formulation

In this section, we introduce notation used in this paper and the formulation of our model in detail.

A. Notation and Formulation

In metric learning problems, we are often given a set of N points $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \mid \mathbf{x}_i \in \Re^d\}$, and a group of triplets, $\mathcal{T} = \{(i, j, l)_\alpha \mid \mathbf{x}_i$ is similar to $\mathbf{x}_j$ but dissimilar to $\mathbf{x}_l, \alpha = 1, \ldots, T\}$, known as side information. Based on these data and the specified side information, we seek a symmetric and positive semi-definite (p.s.d.) matrix $\mathbf{A}$ that parameterizes the Mahalanobis distance function

$$d_{\mathbf{A}}(\mathbf{x}_i, \mathbf{x}_i) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{A} (\mathbf{x}_i - \mathbf{x}_j)}. \tag{1}$$

Note that it is not necessary for all the points in $\mathbf{X}$ to be involved in $\mathcal{T}$. We assume the factorization of $\mathbf{A}$ is $\mathbf{A} = \mathbf{W}\mathbf{W}^T$, where $\mathbf{W}$ is a $d \times r$ matrix with rank r. For simplicity, we use '$\mathbf{x}_{ij}$' to denote '$\mathbf{x}_i - \mathbf{x}_j$' in all the following parts. Overloading notation a bit, we use $\{i, j, l\}$ to index variables corresponding to the data points $\mathbf{x}_i$, $\mathbf{x}_j$ and $\mathbf{x}_l$, respectively.

To express the similarity between points in a compact form, we construct a symmetric adjacency matrix with entries

$$\mathbf{K}_{ij} = \mathbf{K}_{ji} = \begin{cases} 1, & (i, j, \cdot)_\alpha \in \mathcal{T}; \\ 0, & \text{otherwise}; \end{cases} \tag{2}$$

where $(i, j, \cdot)_\alpha$ denotes any triplet involved in $\mathcal{T}$, indicating $\mathbf{x}_i$ and $\mathbf{x}_j$ are a similar pair of points. In the case that the number of triplets is much smaller than the squared number of the data, $\mathbf{K}$ will be a quite sparse matrix. With $\mathbf{K}$, we can derive a loss function parameterized by $\mathbf{A}$ that

$$\begin{aligned}
\mathcal{L}_1(\mathbf{A}) &= \frac{1}{2} \sum_{i,j=1}^{N} (\mathbf{x}_{ij})^T \mathbf{A} (\mathbf{x}_{ij}) \mathbf{K}_{ij} \\
&= \sum_{i,j=1}^{N} (\mathbf{x}_i^T \mathbf{A} \mathbf{x}_i - \mathbf{x}_i^T \mathbf{A} \mathbf{x}_j) \mathbf{K}_{ij} \\
&= \mathrm{tr}(\mathbf{X}^T \mathbf{A} \mathbf{X} \mathbf{D}_{\mathbf{K}}) - \mathrm{tr}(\mathbf{X}^T \mathbf{A} \mathbf{X} \mathbf{K}) \\
&= \mathrm{tr}(\mathbf{X} \mathbf{D}_{\mathbf{K}} \mathbf{X}^T \mathbf{A}) - \mathrm{tr}(\mathbf{X} \mathbf{K} \mathbf{X}^T \mathbf{A}) \\
&= \mathrm{tr}(\mathbf{X} (\mathbf{D}_{\mathbf{K}} - \mathbf{K}) \mathbf{X}^T \mathbf{A}) \\
&= \mathrm{tr}(\mathbf{X} \mathbf{L}_{\mathbf{K}} \mathbf{X}^T \mathbf{A}),
\end{aligned} \tag{3}$$

where $\mathbf{D}_{\mathbf{K}}$ is a diagonal matrix whose diagonal elements are the sums of the row elements of $\mathbf{K}$, and $\mathbf{L}_{\mathbf{K}} = \mathbf{D}_{\mathbf{K}} - \mathbf{K}$ is the Laplacian matrix. As a result, we have encoded all the side information w.r.t. similarity in the loss function $\mathcal{L}_1(\mathbf{A})$.

To punish small distance between dissimilar points, we introduce another loss function $\mathcal{L}_2(\mathbf{A}) = [1 + \mathbf{x}_{ji}^T \mathbf{A} \mathbf{x}_{ji} - \mathbf{x}_{li}^T \mathbf{A} \mathbf{x}_{li}]_+$ within each given triplet $(i, j, l)_\alpha \in \mathcal{T}$, where $\mathcal{L}_2(\mathbf{A})$ is the so called hinge loss w.r.t. $\mathbf{A}$ and $[z]_+ = \max(0, 1 - z)$. This hinge loss is incurred by the dissimilar data whose distance does not exceed, by one unit of distance, the distance between the similar data within the same triplet. For each triplet, we relax the hinge loss by introducing a nonnegative slack variable $\xi_\alpha, \alpha = 1, \ldots, T$:

$$\mathbf{x}_{li}^T \mathbf{A} \mathbf{x}_{li} - \mathbf{x}_{ji}^T \mathbf{A} \mathbf{x}_{ji} \geq 1 - \xi_\alpha. \tag{4}$$

Based on Equation (3) and Equation (4), our model targets to learn a low rank distance function with manifold regularization under similarity and dissimilarity constraints. The model can be formulated as

$$\begin{aligned}
\mathrm{argmin}_{\mathbf{A} \succeq 0, \xi} \quad & \mu \mathrm{rank}(\mathbf{A}) + \tfrac{\lambda}{T} \mathrm{tr}(\mathbf{X} \mathbf{L}_{\mathbf{K}} \mathbf{X}^T \mathbf{A}) + \eta \mathrm{R}_{\mathcal{M}}(\mathbf{A}) \\
& + \textstyle\sum_{t=1}^{T} \xi_\alpha \\
s.t. \quad & \mathbf{x}_{li}^T \mathbf{A} \mathbf{x}_{li} - \mathbf{x}_{ji}^T \mathbf{A} \mathbf{x}_{ji} \geq 1 - \xi_\alpha, \\
& \xi_\alpha \geq 0, \ (i, j, l)_\alpha \in \mathcal{T},
\end{aligned} \tag{5}$$

where $\xi = \{\xi_1, \ldots, \xi_\alpha\}$ is the vector composed of slack variables, μ, λ and η are positive trade-off weights, and $\mathrm{R}_{\mathcal{M}}(\mathbf{A})$ is a manifold regularizer parameterized by $\mathbf{A}$. Introduction of $\mathrm{R}_{\mathcal{M}}(\mathbf{A})$ incorporates the knowledge from unlabeled data into the metric $\mathbf{A}$ by assuming that each transformed $\mathbf{W}^T \mathbf{x}_i$ lies on or close to a low-dimensional manifold. We will discuss this regularizer in more details shortly in Section II-B.

Rank($\mathbf{A}$) is a non-convex function w.r.t. $\mathbf{A}$ and hard to optimize due to the combinatorial nature. Following recent work in matrix completion [15], [16], we replace rank($\mathbf{A}$) with its convex envelop – the nuclear norm of $\mathbf{A}$. This is different from previous l_1-norm sparse regularization in that the element-wise l_1-norm of a matrix does not necessarily generate a low rank. The nuclear norm of $\mathbf{A}$ is defined as the sum of its singular values, i.e., $\| \mathbf{A} \|_* = \sum_{s=1}^{r} \sigma_s(\mathbf{A})$, where σ_s's are the singular values of $\mathbf{A}$, and r is the rank of $\mathbf{A}$. Therefore, the resulting formulation of our model can be written as

$$\begin{aligned}
\mathrm{argmin}_{\mathbf{A} \succeq 0, \xi} \quad & \mu \| \mathbf{A} \|_* + \tfrac{\lambda}{T} \mathrm{tr}(\mathbf{X} \mathbf{L}_{\mathbf{K}} \mathbf{X}^T \mathbf{A}) + \eta \mathrm{R}_{\mathcal{M}}(\mathbf{A}) \\
& + \textstyle\sum_{t=1}^{T} \xi_\alpha \\
s.t. \quad & \mathbf{x}_{li}^T \mathbf{A} \mathbf{x}_{li} - \mathbf{x}_{ji}^T \mathbf{A} \mathbf{x}_{ji} \geq 1 - \xi_\alpha, \\
& \xi_\alpha \geq 0, (i, j, l)_\alpha \in \mathcal{T}.
\end{aligned} \tag{6}$$

B. Parametric Manifold Regularization

We assume all the transformed data $\mathbf{W}^T \mathbf{x}_i$'s lie on or close to a low dimensional manifold and use the projection distance from this manifold to regularize our learning model. We approximately calculate the (squared) projection distance of $\mathbf{W}^T \mathbf{x}_i$ onto the manifold $\mathcal{M}$ as

$$\mathrm{P}_{\mathcal{M}}(\mathbf{A}, \mathbf{x}_i) = \|\mathbf{W}^T \mathbf{x}_i - \sum_{j \in \mathcal{N}_i} m_{ij} \mathbf{W}^T \mathbf{x}_j\|_2^2, \tag{7}$$

where $\mathcal{N}_i$ is the set of k-nearest neighbors of $\mathbf{W}^T\mathbf{x}_i$, m_{ij}'s are the locally linear reconstruction weights with $\sum_{j\in\mathcal{N}_i} m_{ij} = 1$, and $m_{ij} \geq 0$. Following the works by [17] and [18] we have

Theorem 1: In the limit, as the number of data points increase, $\mathrm{P}_{\mathcal{M}}(\mathbf{A}, \mathbf{x}_i)$ is an orthogonal projection distance of $\mathbf{W}^T\mathbf{x}_i$ onto the principal surface, $\mathcal{M}$, of the transformed data points.

Since the space limitation, we omit the proof of this theorem here.

We define

$$\mathrm{P}_{\mathcal{M}}(\mathbf{A}) = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{W}^T\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{W}^T\mathbf{x}_j\|_2^2, \tag{8}$$

and denote $\mathbf{M} = \{m_{ij}\}^{N\times N}$ as the reconstruction weights matrix with m_{ij} as its elements at the i-th row and j-th column. By simply calculation we have

$$\begin{aligned} & \|\mathbf{W}^T\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{W}^T\mathbf{x}_j\|_2^2 \\ = \ & \|\sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{W}^T\mathbf{x}_{ij}\|_2^2 \\ = \ & \sum_{j,l\in\mathcal{N}_i} m_{ij}m_{il}\mathbf{x}_{ij}^T\mathbf{A}\mathbf{x}_{il} \\ = \ & \sum_{j,l\in\mathcal{N}_i} m_{ij}m_{il}\mathbf{G}_{jl} \end{aligned} \tag{9}$$

where $\mathbf{G}_{jl} = \mathbf{x}_{ij}^T\mathbf{A}\mathbf{x}_{il} = \mathrm{tr}(\mathbf{x}_{il}\mathbf{x}_{ij}^T\mathbf{A})$. Substituting $\mathrm{P}_{\mathcal{M}}(\mathbf{A})$ for the manifold regularizer in Problem (6), we obtain

$$\begin{aligned} \mathrm{argmin}_{\mathbf{A}\succeq 0,\mathbf{M},\xi} \quad & \mu \parallel \mathbf{A} \parallel_* + \frac{\lambda}{T}\mathrm{tr}(\mathbf{X}\mathbf{L}_{\mathbf{K}}\mathbf{X}^T\mathbf{A}) \\ & + \frac{\eta}{N}\sum_{i=1}^{N}\sum_{j,l\in\mathcal{N}_i} m_{ij}m_{il}\mathbf{G}_{jl} + \sum_{t=1}^{T}\xi_\alpha \\ s.t. \quad & \mathbf{x}_{li}^T\mathbf{A}\mathbf{x}_{li} - \mathbf{x}_{ji}^T\mathbf{A}\mathbf{x}_{ji} \geq 1 - \xi_\alpha, \\ & \xi_\alpha \geq 0, (i,j,l)_\alpha \in \mathcal{T}, \\ & \sum_{j\in\mathcal{N}_i} m_{ij} = 1, m_{ij} \geq 0. \end{aligned} \tag{10}$$

One can see that the third term in the objective function of Problem (10) is a manifold regularizer parameterized by $\mathbf{M}$. We will show below that Problem (10) is convex w.r.t. $\mathbf{A}$ and $\mathbf{M}$ respectively, by proving that each term in the objective function and each constraint is convex w.r.t. them respectively.

Theorem 2: Problem (10) is convex w.r.t. $\mathbf{A}$ and $\mathbf{M}$ respectively.

Proof: First, it is easy to verify that all the terms in the objective function and constraints are convex w.r.t. $\mathbf{A}$. Second, since $\mathbf{G}$ is a p.s.d. matrix, the third term in the objective function is convex w.r.t. $\mathbf{M}$. Hence, we can get the conclusion that Problem (10) is convex w.r.t. $\mathbf{A}$ and $\mathbf{M}$ respectively. ■

Since Problem (10) is convex w.r.t. $\mathbf{A}$ and $\mathbf{M}$ respectively, we develop an alternating optimization algorithm to solve it, which proves effective to find a satisfactory solution [19].

Algorithm 1 LRMLMR

```
1:  Input:
2:      $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$; $\mathcal{T} = \{(i,j,l)_\alpha\}$; $\mu$, $\lambda$, $\eta$; $\mathbf{A}^0$;
3:  Output:
4:      $\mathbf{A}$, $\mathbf{M}$;
5:  Steps:
6:  Initial $\mathbf{A}$ using $\mathbf{A}^0$;
7:  while t < Maxit and not converge do
8:      1) For fixed $\mathbf{A}^t$, solve $\mathbf{M}^t$ from Problem (11);
9:      2) For fixed $\mathbf{M}^t$, solve $\mathbf{A}^{t+1}$ from Problem (10);
10: end while
```

III. Optimization

The proposed optimization algorithm is shown in Algorithm 1. We call it LRMLMR. LRMLMR begins with initializing $\mathbf{A}$ using a $d \times d$ matrix, e.g., the identity matrix or the inverse of the covariance matrix of the data. Then it learns $\mathbf{M}$ and $\mathbf{A}$ alternatingly. More precisely, when $\mathbf{A}$ is fixed, LRMLMR learns $\mathbf{M}$ by solving the problem as below:

$$\begin{aligned} \mathrm{argmin}_{\mathbf{M}} \quad & \sum_{i=1}^{N} \|\mathbf{W}^T\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{W}^T\mathbf{x}_j\|_2^2 \\ s.t. \quad & \sum_{j\in\mathcal{N}_i} m_{ij} = 1, m_{ij} \geq 0. \end{aligned} \tag{11}$$

Furthermore, this problem can be decomposed into N independent problems for each $i \in \{1, \ldots, N\}$,

$$\begin{aligned} \mathrm{argmin}_{\mathbf{M}_{i*}} \quad & \|\mathbf{W}^T\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{W}^T\mathbf{x}_j\|_2^2 \\ s.t. \quad & \sum_{j\in\mathcal{N}_i} m_{ij} = 1, m_{ij} \geq 0, \end{aligned} \tag{12}$$

where $\mathbf{M}_{i*}$ is the i-th row of $\mathbf{M}$. Problem (12) is a standard quadratic programming problem and can be solved with classic algorithms and online softwares. Using the Lagrange multiplier, Problem (12) can be rewritten as

$$\begin{aligned} \mathrm{argmin}_{\mathbf{M}_{i*}} \quad & \|\mathbf{W}^T\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{W}^T\mathbf{x}_j\|_2^2 + \pi(\sum_{j\in\mathcal{N}_i} m_{ij} - 1) \\ s.t. \quad & m_{ij} \geq 0, \end{aligned} \tag{13}$$

where π is a positive trade-off constant. Since all the m_{ij} are nonnegative, i.e., $m_{ij} \geq 0$, the second term $\pi(\sum_{j\in\mathcal{N}_i} m_{ij} - 1)$ is equivalent to an l_1-norm regularizer to $\mathbf{M}_{i*}$, which can encourage the sparsity of $\mathbf{M}_{i*}$ [20], [21]. In other words, this algorithm could adaptively choose the meaningful neighbors to reconstruct each datum $\mathbf{W}^T\mathbf{x}_i$.

For a given $\mathbf{M}$, LRMLMR learns $\mathbf{A}$ using a fixed-point iterative method. In the t-th iteration, the fixed-point iterative method involves two alternating steps:

1) (gradient step) $\quad \mathbf{Z}^t = \mathbf{A}^t - \tau g(\mathbf{A}^t)$,
2) (shrinkage step) $\quad \mathbf{A}^{t+1} = S_{\tau\mu}(\mathbf{Z}^t)$.

In the gradient step, $g(\mathbf{A}^t)$ is the gradient of the objective function in Problem (10) w.r.t. $\mathbf{A}^t$ (excluding the nuclear norm term), and τ is the step size. Following [22], we can express ξ_α as a function of $\mathbf{A}$:

$$\xi_\alpha(\mathbf{A}) = [1 + \mathbf{x}_{ji}^T\mathbf{A}\mathbf{x}_{ji} - \mathbf{x}_{li}^T\mathbf{A}\mathbf{x}_{li}]_+, \forall (i,j,l)_\alpha \in \mathcal{T} \tag{14}$$

where $[z]_+ = \max(0, 1-z)$ is the hinge loss. Note that the hinge loss is not differentiable, but we can compute its subgradient and use a standard descent algorithm to optimize the problem. Thus we can calculate $g(\mathbf{A}^t)$ as

$$\begin{aligned} g(\mathbf{A}^t) &= \frac{\lambda}{T}\mathbf{X}\mathbf{L}_{\mathbf{K}}\mathbf{X}^T + \frac{\eta}{N}\sum_{i=1}^{N}\sum_{j,l\in\mathcal{N}_i} m_{ij}m_{il}\mathbf{x}_{il}\mathbf{x}_{ij}^T \\ &+ \sum_{(i,j,l)_\alpha\in\mathcal{S}} (\mathbf{x}_{ji}\mathbf{x}_{ji}^T - \mathbf{x}_{li}\mathbf{x}_{li})^T \end{aligned} \quad (15)$$

where $\mathcal{S} \subset \mathcal{T}$ is the set of triplets whose corresponding slack variable exceeds zero, $\xi_\alpha(\mathbf{A}^t) > 0$.

In the shrinkage step, $S_{\tau\mu}(\mathbf{Z}^t)$ is a matrix shrinkage operator on $\mathbf{Z}^t$. Since $\mathbf{Z}^t$ is a symmetric and p.s.d. matrix, we adapt the eigenvalue decomposition method to shrink the rank of $\mathbf{Z}$. Let $\mathbf{Z}^t = \mathbf{U}\Lambda\mathbf{U}^T$ is the eigenvalue decomposition of $\mathbf{Z}^t$. Then $S_{\tau\mu}(\mathbf{Z}^t) = \mathbf{U}\max\{\Lambda - \tau\mu, \mathbf{0}\}\mathbf{U}^T$, where max is elementwise. That is, the shrinkage operator shifts the eigenvalues down, and truncates any eigenvalue less than $\tau\mu$ to zero. This step reduces the nuclear norm as well.

For the convergence properties of this fixed-point iterative algorithm, we present a theorem as below.

***Theorem** 3:* For a fixed $\mathbf{M}$, the sequence $\{\mathbf{A}^t\}$ generated by the fixed-point iterations with $\tau \in (0, 2/\lambda_{max}(g_{\mathbf{M}}(A)))$ converges to some $\mathbf{A}^* \in \mathcal{A}^*$, where $g_{\mathbf{M}}(A)$ is the gradient of the objective function in Problem (10) w.r.t. $\mathbf{A}$ for a fixed $\mathbf{M}$ (excluding the nuclear norm term), $\lambda_{max}(g_{\mathbf{M}}(A))$ is the maximum eigenvalue of $g_{\mathbf{M}}(A)$ and $\mathcal{A}^*$ is the set of optimal solutions of Problem (10).
The proof of this theorem is similar to that of theorem 4 in [23]. Due to the space limitation, we omit it here.

IV. A Fast Solver

The optimization problem (10) is not jointly convex w.r.t. $\mathbf{A}$ and $\mathbf{M}$. Even though we can employ an alternating optimization algorithm to solve it, the speed of implementation still hampers its application to real data. In this case, a typical workaround is to minimize a surrogate convex loss function which upper bounds the original non-convex one [24], [25]. Here, motivated by the works of [24] and [25], we derive a tight upper bound on the objective function in Problem (11), which can result in a convex nonparametric manifold regularizer to the original optimization problem (10). Especially, based on this new manifold regularizer, we can find a fast solver to Problem (10) directly. Before introducing the detailed convex relaxation procedure, we first present a lemma as below:

***Lemma** 1:* There exists a number $\vartheta \geq 0$, to guarantee $\|\mathbf{W}^T\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{W}^T\mathbf{x}_j\|_2^2 \leq \vartheta\|\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{x}_j\|_2^2$, where $\sum_{j\in\mathcal{N}_i} m_{ij} = 1, m_{ij} \geq 0$, and $\mathbf{W}\mathbf{W}^T = \mathbf{A}$.

Proof: Suppose the eigenvalue decomposition of $\mathbf{A}$ is $\mathbf{A} = \mathbf{V}\Lambda_{\mathbf{A}}\mathbf{V}^T$, and $\lambda_{\mathbf{A}}^1$ is the maximum eigenvalue of $\mathbf{A}$. Then we have

$$\begin{aligned} &\|\mathbf{W}^T\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{W}^T\mathbf{x}_j\|_2^2 \\ = &\sum_{j,l\in\mathcal{N}_i} m_{ij}m_{il}\mathbf{x}_{ij}^T\mathbf{A}\mathbf{x}_{il} \\ \leq &\ \lambda_{\mathbf{A}}^1\|\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{x}_j\|_2^2. \end{aligned} \quad (16)$$

Let $\vartheta = \lambda_{\mathbf{A}}^1$ and substitute it in Equation (16). We can get the result presented in the above theorem. ■

Based on this lemma, we have

***Theorem** 4:* we can find a tight upper bound on the objective function in Problem (11), such that $\sum_{i=1}^N \|\mathbf{W}^T\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{W}^T\mathbf{x}_j\|_2^2 \leq \tilde{\vartheta}\sum_{i=1}^N \|\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{x}_j\|_2^2$, where $\tilde{\vartheta}$ is a positive constant, $\sum_{j\in\mathcal{N}_i} m_{ij} = 1, m_{ij} \geq 0, and\ \mathbf{W}\mathbf{W}^T = \mathbf{A}$.
Therefore, we can integrate $\sum_{i=1}^N \|\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} m_{ij}\mathbf{x}_j\|_2^2$ into the original learning model as a nonparametric manifold regularizer, instead of the parametric one in Problem (10). As a result, this updated model can be solved rather faster than the original one: we directly learn $\mathbf{M}$ from the given data, and then learn $\mathbf{A}$ based upon $\mathbf{M}$ using a fixed-point iterative algorithm. Since this updated problem is still convex w.r.t. $\mathbf{A}$, we can obtain a global optimum for this new model, and meantime guarantee that it is also an approximated optimal solution for the original problem, with high confidence.

V. Experiments

To evaluate our proposed method, LRMLMR, we conduct experiments over 12 standard UCI data sets. The properties of each data set are shown in the 2nd-4th columns of Table I. For all the data sets, we scale the features to $[-1, +1]$. No other preprocessing is used. We compare LRMLMR to the naive Euclidean distance (Euclidean) method, Mahalanobis distance method with the inverse of the covariance matrix (InvCov), and three state-of-the-art metric learning methods, i.e., large margin nearest neighbor (LMNN) [2], information theoretic metric learning (ITML) [5] and sparse metric learning (SML) [26].

Following previous work, e.g., [26], we generate triplets using the label information. More specifically, given a randomly chosen triplet $\{i, j, l\}$ sampled from the training data, if the first two data samples share the same label and the third one has a different label, we incorporate this triplet in the triplet set $\mathcal{T}$. Namely, for a triplet of data $\{\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_l\}$, $\mathbf{x}_i$ and $\mathbf{x}_j$ are similar, while $\mathbf{x}_i$ and $\mathbf{x}_l$ are dissimilar. For LMNN, the nearest neighbor points for each training sample need to be provided before training. In order to provide equal side information, for each training sample, we regard their similar samples appearing in the similar pairs as their nearest neighbors. For ITML, two sets need to be used for evaluating the algorithm: a similar set $\mathcal{T}_1$ containing the similar pairs and a dissimilar set $\mathcal{T}_2$ containing the dissimilar pairs. In

Table I

DATA SETS (T: TRAINING DATA; A: ATTRIBUTES; C: CLASSES) AND TEST ACCURACY.

Data sets	♯ of T	♯ of A	♯ of C	Euclidean	InvCov	LMNN	ITML	SML	LRMLMR
Balance	625	4	3	0.7166	0.7017	0.7991	0.8594	0.7944	**0.8609**
Dermatology	366	34	6	0.9524	0.6420	**0.9786**	0.9751	0.9732	0.9778
Ecoli	336	8	8	0.9109	0.8625	0.8946	0.9119	**0.9226**	0.9218
Glass	214	9	7	0.7154	0.6974	0.7088	0.7547	0.7481	**0.7571**
Iris	150	4	3	0.9389	0.7781	0.9532	**0.9736**	0.9683	0.9735
Optdigits	5620	64	10	0.8835	0.6536	0.9210	0.9395	0.9269	**0.9445**
Pima	768	8	2	0.6002	0.5916	0.5966	0.6263	0.6422	**0.6483**
Protein	116	20	6	0.7737	0.7294	0.8273	0.8274	**0.8567**	0.8562
Segmentation	2310	19	7	0.8808	0.7666	0.8870	0.8865	0.9494	**0.9510**
Soybean	47	35	4	0.9778	0.7932	0.9890	**0.9983**	0.9824	0.9976
Thyroid	215	5	3	0.9057	0.8327	**0.9440**	0.9329	0.9352	0.9439
Wine	178	13	3	0.9069	0.6884	0.9697	0.9592	0.9677	**0.9720**

a similar manner to [26], we generate these two sets from the triplet set $\mathcal{T}$ by placing (i, j) into the similar set $\mathcal{T}_1$ and (i, l) into the dissimilar set $\mathcal{T}_2$, provided (i, j, l) is a triplet in $\mathcal{T}$. As discussed in [26], such a strategy provides a fair level of supervision for the comparison methods, in the sense that roughly the same information is presented to different methods.

We use the same ratio, i.e., 0.85:0.15 as used by [26], to split the data sets into a training set and a test set. From the same data set, 1500 triplets are generated from the training set based on the strategy mentioned above, while 1000 triplets are sampled from the test set. During testing, if the distance between two similar data in the triplet is not greater than the distance between the dissimilar pair (calculated based on the learned metric), this triplet is regarded as classified correctly. We count the ratio of correctly classified triplets to give the final accuracy score. This procedure was used for all five methods. The final results are given as an average over 20 random splits of the data. We use codes of LMNN and ITML by the authors from their personal websites. The trade-off parameter γ used in ITML and SML, λ and η used in LRMLMR are tuned in the range $\{0.001, 0.01, 0.1, 1, 10\}$ by cross validation. The trade-off weights used in LMNN is tuned in the range $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ by cross validation. The parameters μ and τ used in LRMLMR are fixed as 1 and 0.001, respectively. The number of nearest neighbors, k, for locally linear reconstruction in LRMLMR is set to 12, empirically. The maximum number of iterations for LMNN, ITML and the fixed-point iterative method used in LRMLMR is fixed to 10000. Since LRMLMR and its fast version basically achieve similar results, we only report the results obtained by the fast version in the following experiments.

We show the classification accuracy in the 5th-10th columns of Table I, where the best one for each data set is highlighted in bold. From Tabel I, we can get some interesting observations. Firstly, since we have scaled the data features to $[-1, +1]$, Euclidean distance can perform much better than the InvCov Mahalanobis distance. Secondly, in most cases, metric learning methods can outperform both Euclidean distance and InvCov Mahalanobis distance. Initialized with the identity matrix, both ITML and LRMLMR greatly improves the performance of the Euclidean distance over all the data sets. However, LMNN performs slightly worse than the Euclidean distance over some data sets. Finally, since LRMLMR utilizes both label information from labeled data and geometric information from unlabeled data, it achieves the overall best performance among other state-of-the-art metric learning approaches. This also demonstrates the advantages of our method to learn a low rank distance metric with manifold regularization.

Ranks of the learned distance matrices via each metric learning method are shown in Figure 1. One can see that the ranks of the distance matrices learned by LRMLMR are almost consistently lower than that learned by other methods over all the data sets. The phenomenon is more evident, especially when compared to Euclidean distance method, InvCov Mahalanobis distance method, ITML and SML. SML can learn a sparse distance metric. However, we can see that the ranks of the distance metric learned by SML is of full rank but not of a low rank. Although LMNN can impose partial sparsity, its formulation does not target sparsity directly. As can be seen, LMNN learns a low rank distance metric over some data sets. However, for most of the data sets, the ranks of the distance matrices learned by LMNN are much higher than that learned by LRMLMR, e.g., on the Pima, Protein, Segmentation and Soybean data sets. Moreover, as discussed above, LRMLMR achieves much better performance than LMNN almost over all the data sets.

VI. CONCLUSION

In this paper, we presented a semi-supervised method to learn a low rank Mahalanobis distance function from data. By approximating the projection distance from a manifold, we introduced a novel parametric manifold regularizer for the metric learning. This takes advantages of the important geometric information from data and proves beneficial for boosting the performance of learned metric. In addition, our method focused on learning a low-rank metric directly. This distinguishes our method from many previous approaches. The resulting configuration is convex with respect to the manifold structure and the distance function, respectively. We solved it with an alternating optimization algorithm,

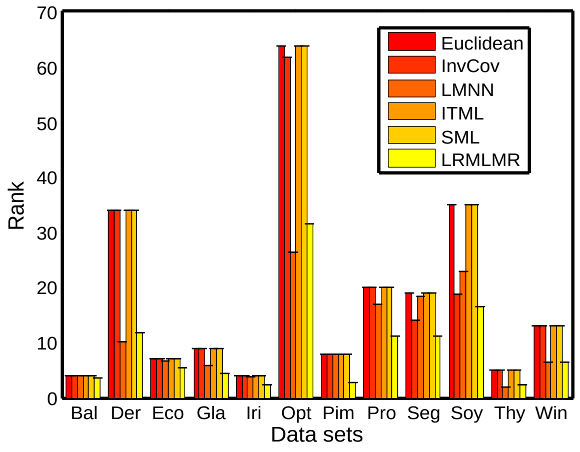

Figure 1. Ranks of distance matrices.

which is guaranteed to find a local optimal solution. For efficient implementation, we further presented a fast algorithm for the original problem, in which the manifold structure and the distance function are learned independently without alternating minimization. Experimental results over 12 standard UCI data sets demonstrate that our method can usually outperform other metric learning approaches.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (NSFC) under grants No. 60825301 and No. 61075052.

References

[1] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. J. Russell, “Distance Metric Learning, with Application to Clustering with Side-Information,” in *NIPS*, 2002, pp. 505–512.

[2] K. Q. Weinberger, J. Blitzer, and L. K. Saul, “Distance Metric Learning for Large Margin Nearest Neighbor Classification,” in *NIPS*, 2005, pp. 1473–1480.

[3] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov, “Neighbourhood Components Analysis,” in *NIPS*, 2005, pp. 513–520.

[4] A. Globerson and S. Roweis, “Metric Learning by Collapsing Classes,” in *NIPS*, 2006, pp. 451–458.

[5] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, “Information-Theoretic Metric Learning,” in *ICML*, 2007, pp. 209–216.

[6] S. C. H. Hoi, W. Liu, and S.-F. Chang, “Semi-Supervised Distance Metric Learning for Collaborative Image Retrieval,” in *CVPR*, 2008, pp. 1–7.

[7] G.-J. Qi, J. Tang, Z.-J. Zha, T.-S. Chua, and H.-J. Zhang, “An Efficient Sparse Metric Learning in High-Dimensional Space via l_1-penalized Log-determinant Regularization,” in *ICML*, 2009, pp. 841–848.

[8] Y. Ying, K. Huang, and C. Campbell, “Sparse Metric Learning via Smooth Optimization,” in *NIPS*, 2009, pp. 2214–2222.

[9] T. Hastie and R. Tibshirani, “Discriminant Adaptive Nearest Neighbor Classification,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 6, pp. 607–616, 1996.

[10] R. Jin, S. Wang, and Y. Zhou, “Regularized Distance Metric Learning:Theory and Algorithm,” in *NIPS*, 2009, pp. 862–870.

[11] J. Chen, Z. Zhao, J. Ye, and H. Liu, “Nonlinear adaptive distance metric learning for clustering,” in *SIGKDD*, 2007, pp. 123–132.

[12] J. B. Tenenbaum, V. d. Silva, and J. C. Langford, “A Global Geometric Framework for Nonlinear Dimensionality Reduction,” *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[13] S. T. Roweis and L. K. Saul, “Nonlinear Dimensionality Reduction by Locally Linear Embedding,” *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[14] M. S. Baghshah and S. B. Shouraki, “Semi-Supervised Metric Learning Using Pairwise Constraints,” in *IJCAI*, 2009, pp. 1217–1222.

[15] E. J. Candès and T. Tao, “The Power of Convex Relaxation: Near-Optimal Matrix Completion,” *IEEE Trans. Inf. Theory,*, vol. 56, no. 5, pp. 2053–2080, 2010.

[16] E. J. Candès and B. Recht, “Exact Matrix Completion via Convex Optimization,” *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, 2009.

[17] T. Hastie, “Principal Curves and Surfaces,” Ph.D. dissertation, Stanford University, 1984.

[18] S. Gerber, T. Tasdizen, and R. Whitaker, “Dimensionality Reduction and Principal Surfaces via Kernel Map Manifolds,” in *ICCV*, 2009, pp. 529–536.

[19] D. P. Bertsekas, *Nonlinear Programming (Second Edition)*. Athena Scientific, 1999.

[20] D. L. Donoho, “Compressed Sensing,” *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[21] T. Hastie, R. Tibshirani, and J. Friedman, *Elements of Statistical Learning: Data Mining, Inference and Prediction (Second Edition)*, 2nd ed. New York: Springer-Verlag, February 2009.

[22] K. Q. Weinberger and L. K. Saul, “Fast Solvers and Efficient Implementations for Distance Metric Learning,” in *ICML*, 2008, pp. 1160–1167.

[23] S. Ma, D. Goldfarb, and L. Chen, “Fixed Point and Bregman Iterative Methods for Matrix Rank Minimization,” *Mathematical Programming*, vol. 128, pp. 321–353, 2011.

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

[25] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, “Large Margin Methods for Structured and Interdependent Output Variables,” *Journal of Machine Learning Research*, vol. 6, pp. 1453–1484, 2005.

[26] R. Rosales and G. Fung, “Learning Sparse Metrics via Linear Programming,” in *KDD*, 2006, pp. 367–373.