Reduced-Rank Local Distance Metric Learning

Yinjie Huang[1], Cong Li[1], Michael Georgiopoulos[1], and Georgios C. Anagnostopoulos[2]

[1] University of Central Florida, Department of Electrical Engineering & Computer Science, 4000 Central Florida Blvd, Orlando, Florida, 32816, USA
yhuang@eecs.ucf.edu, licong1112@gmail.com, michaelg@ucf.edu
[2] Florida Institute of Technology, Department of Electrical and Computer Engineering, 150 W University Blvd, Melbourne, FL 32901, USA
georgio@fit.edu

Abstract. We propose a new method for local metric learning based on a conical combination of Mahalanobis metrics and pair-wise similarities between the data. Its formulation allows for controlling the rank of the metrics' weight matrices. We also offer a convergent algorithm for training the associated model. Experimental results on a collection of classification problems imply that the new method may offer notable performance advantages over alternative metric learning approaches that have recently appeared in the literature.

Keywords: Metric Learning, Local Metric, Proximal Subgradient Descent, Majorization Minimization.

1 Introduction

Many Machine Learning problems and algorithms entail the computation of distances with prime examples being the k-nearest neighbor (KNN) decision rule for classification and the k-Means algorithm for clustering problems. Also, when computing distances, the use of the Euclidean distance metric, or a weighted variation of it, the Mahalanobis metric, are most often encountered because of their simplicity and geometric interpretation. However, employing these metrics for computing distances may not necessarily perform well for all problems. Early on, attention was directed to data-driven approaches in order to infer the best metric for a given problem (*e.g.* [1] and [2]). This is accomplished by taking advantage of the data's distributional characteristics or other *side information*, such as similarities between samples. In general, such paradigms are referred to as *metric learning* techniques. A typical instance of such approaches is the learning of the weight matrix that determines the Mahalanobis metric. This particular task can equivalently be viewed as learning a decorrelating linear transformation of the data in their native space and computing Euclidean distances in the range space of the learned linear transform (feature space). When the problem at hand is a classification problem, a KNN algorithm based on the learned metric is eventually employed to label samples.

H. Blockeel et al. (Eds.): ECML PKDD 2013, Part III, LNAI 8190, pp. 224–239, 2013.

This paper focuses on metric learning methods for classification tasks, where the Mahalanobis metric is learned with the assistance of pair-wise sample similarity information. In our context, two samples will be deemed similar, if they feature the same class label. The goal of such approaches is to map similar samples close together and to map dissimilar samples far apart as measured by the learned metric. This is done so that an eventual application of a KNN decision rule exhibits improved performance over an application of KNN using a Euclidean metric.

Many such algorithms show significant improvements over the case of KNN that uses Euclidean metrics. For example, [1] poses similarity-based metric learning as a convex optimization problem, while [3] builds a trainable system to map similar faces to low dimensional spaces using a convolutional network to address geometric distortions. Moreover, [2] provides an online algorithm for learning a Mahalanobis metric based on kernel operators. Another approach, Neighborhood Components Analysis (NCA) [4], maximizes the leave-one-out performance on the training data based on stochastic nearest neighbors. Furthermore, in Large Margin Nearest Neighbor (LMNN) [5], the metric is learned so that the k-nearest neighbors of each sample belong to the same class, while others are separated by a large margin. Finally, [6] formulates the problem using information entropy and proposes the Information Theoretic Metric Learning (ITML) technique. In specific, ITML minimizes the differential relative entropy between two multivariate Gaussian distributions with distance metric constraints.

A common thread of the aforementioned methods is the use of a single, global metric, *i.e.*, a metric that is used for all distance computations. However, learning a global metric may not be well-suited in some settings that entail multi-modality or non-linearities in the data. To illustrate this point, Figure 1 displays a toy dataset consisting of 4 samples drawn from two classes. Sub-figure (a) shows the samples in their native space and sub-figure (b) depicts their images in the feature space resulting from learning a global metric. Finally, sub-figure (c) depicts the transformed data, when a local metric is learned, that takes into account the location and similarity characteristics of the data involved. We'll refer to such metrics as *local metrics*. Unlike the results obtained via the use of a global metric, one can (somewhat, due to the 3-dimensional nature of the depiction) observe in sub-figure (c) that images of similar samples (in this case, of the same class label) have been mapped closer to each other, when a local metric is learned. This may potentially result into improving 1-NN classification performance, when compared to the sample distributions in the other two cases.

Much work has been already performed on local metric learning. For example, [7] defines "local" as nearby pairs. In particular, they develop a model that aims to co-locate similar pairs and to separate dissimilar pairs. Additionally, their probabilistic framework is solved using an Expectation-Maximization-like algorithm. [8] learns local metrics through reducing neighborhood distances in directions that are orthogonal to the local decision boundaries, while expanding those parallel to the boundaries. In [9], the authors of LMNN also developed the LMNN-Multiple Metric (LMNN-MM) technique. When LMNN-MM is applied

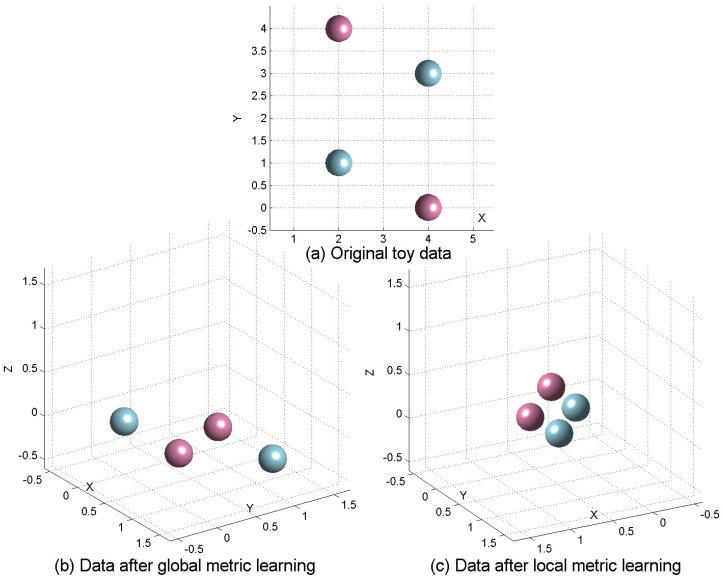

Fig. 1. Toy dataset that illustrates the potential advantages of learning a local metric instead of a global one. (a) Original data distribution. (b) Data distribution in the feature space obtained by learning a global metric. (c) Data distribution in the feature space obtained by learning a local metric.

in a classification context, the number of metrics utilized equals the number of classes. [10] introduced a similar approach, in which a metric is defined for each cluster. Moreover, in [11], the authors proposed Generative Local Metric Learning (GLML), which learns local metrics through NN classification error minimization. Their model employs a rather strong assumption, namely, they assume that the data has been drawn from a Gaussian mixture. Furthermore, in [12], the authors propose Parametric Local Metric Learning (PLML), in which each local metric is defined in relation to an anchor point of the instance space. Next, they use a linear combination of the resulting metric-defining weight matrices and employ a projected gradient method to optimize their model.

In this paper, we propose a new local metric learning approach, which we will be referring to as Reduced-Rank Local Metric Learning (R^2LML). As detailed in Section 2, for our method, the local metric is modeled as a conical combination of Mahalanobis metrics. Both the Mahalonobis metric weight matrices and the coefficients of the combination are learned from the data with the aid of pair-wise similarities in order to map similar samples close to each other and dissimilar samples far from each other in the feature space. Furthermore, the proposed problem formulation is able to control the rank of the involved linear mappings through a sparsity-inducing matrix norm. Additionally, in Section 3 we supply an algorithm for training our model. We then show that the set of fixed points of our

algorithm includes the Karush-Kuhn-Tucker (KKT) points of our minimization problem. Finally, in Section 4 we demonstrate the capabilities of R^{2}LML with respect to classification tasks. When compared to other recent global or local metric learning methods, R^{2}LML exhibits the best classification accuracy in 7 out of the 9 datasets we considered.

2 Problem Formulation

Let $\mathbb{N}_M \triangleq \{1, 2, \ldots, M\}$ for any positive integer M. Suppose we have a training set $\{\boldsymbol{x}_n \in \mathbb{R}^D\}_{n\in\mathbb{N}_N}$ and corresponding pair-wise sample similarities arranged in a matrix $\boldsymbol{S} \in \{0,1\}^{N\times N}$ as side information with the convention that, if $\boldsymbol{x}_m$ and $\boldsymbol{x}_n$ are similar, then $s_{mn} = 1$; if otherwise, then $s_{mn} = 0$. In a classification scenario, two samples can be naturally deemed similar (or dissimilar), if they feature the same (or different) class labels.

Now, the Mahalanobis distance between two samples $\boldsymbol{x}_n$ and $\boldsymbol{x}_m$ is defined as $d_{\boldsymbol{A}}(\boldsymbol{x}_m, \boldsymbol{x}_n) \triangleq \sqrt{(\boldsymbol{x}_m - \boldsymbol{x}_n)^T \boldsymbol{A}(\boldsymbol{x}_m - \boldsymbol{x}_n)}$, where $\boldsymbol{A} \in \mathbb{R}^{D\times D}$ is a positive semi-definite matrix (denoted as $\boldsymbol{A} \succeq 0$), which we will refer to as the *weight matrix of the metric*. Obviously, when $\boldsymbol{A} = \boldsymbol{I}$, the previous metric becomes the usual Euclidean distance. Being positive semi-definite, the weight matrix can be expressed as $\boldsymbol{A} = \boldsymbol{L}^T\boldsymbol{L}$, where $\boldsymbol{L} \in \mathbb{R}^{P\times D}$ with $P \leq D$. Hence, the previously defined distance can be expressed as $d_{\boldsymbol{A}}(\boldsymbol{x}_m, \boldsymbol{x}_n) = \|\boldsymbol{L}(\boldsymbol{x}_m - \boldsymbol{x}_n)\|_2$. Evidently, this last observation implies that the Mahalanobis distance based on $\boldsymbol{A}$ between two points in the native space can be viewed as the Euclidean distance between the images of these points in a feature space obtained through the linear transformation $\boldsymbol{L}$.

In metric learning, we are trying to learn $\boldsymbol{A}$ so to minimize the distances between pairs of similar points, while maintaining above a certain threshold (if not maximizing) the distances between dissimilar points in the feature space. Such a problem could be formulated as follows:

$$\begin{aligned} \underset{\boldsymbol{A}\succeq 0}{min} \ & \sum_{m,n} s_{mn} d_{\boldsymbol{A}}(\boldsymbol{x}_m, \boldsymbol{x}_n) \\ s.t. \ & \sum_{m,n} (1 - s_{mn}) d_{\boldsymbol{A}}(\boldsymbol{x}_m, \boldsymbol{x}_n) \geq 1 \end{aligned} \tag{1}$$

Problem (1) is a semi-definite programming problem involving a global metric based on $\boldsymbol{A}$. There are several methods for learning a single global metric like the ones used for LMNN, ITML and NCA. However, as we have shown in Figure 1, use of a global metric may not be advantageous under all circumstances.

In this paper, we propose R^{2}LML, a new local metric approach, which we delineate next. Our formulation assumes that the metric involved is expressed as a conical combination of $K \geq 1$ Mahalanobis metrics. We also define a vector $\boldsymbol{g}^k \in \mathbb{R}^N$ for each local metric k. The n^{th} element g_n^k of this vector may be regarded as a measure of how important metric k is, when computing distances involving the n^{th} training sample. We constrain the vectors $\boldsymbol{g}^k$ to belong to $\Omega_g \triangleq$

$\left\{\{\boldsymbol{g}_k\}_{k\in\mathbb{N}_K} \in [0,1]^N : \boldsymbol{g}^k \succeq \mathbf{0},\ \sum_k \boldsymbol{g}^k = \mathbf{1}\right\}$, where '$\succeq$' denotes component-wise ordering. The fact that the $\boldsymbol{g}^k$'s need to sum up to the all-ones vector $\mathbf{1}$ forces at least one metric to be relevant, when computing distances from each training sample. Note that, if $K = 1$, $\boldsymbol{g}^1 = \mathbf{1}$, which corresponds to learning a single global metric.

Based on what we just described, the weight matrix for each pair (m, n) of training samples is given as $\sum_k \boldsymbol{A}^k g_m^k g_n^k$. Observe that the distance between every pair of points features a different weight matrix. Motivated by Problem (1), one could consider the following formulation:

$$\begin{aligned} \underset{\boldsymbol{L}^k, \boldsymbol{g}^k \in \Omega_g, \xi^k_{m,n} \geq 0}{min} \quad & \sum_k \sum_{m,n} s_{mn} \left\| \boldsymbol{L}^k \Delta \boldsymbol{x}_{mn} \right\|_2^2 g_n^k g_m^k + \\ & + C \sum_k \sum_{m,n} (1 - s_{mn}) \xi^k_{mn} + \lambda \sum_k \text{rank}(\boldsymbol{L}^k) \\ s.t. \quad & \left\| \boldsymbol{L}^k \Delta \boldsymbol{x}_{mn} \right\|_2^2 \geq 1 - \xi^k_{mn}, \quad m, n \in \mathbb{N}_N,\ k \in \mathbb{N}_K \end{aligned} \tag{2}$$

where $\Delta \boldsymbol{x}_{mn} \triangleq \boldsymbol{x}_m - \boldsymbol{x}_n$ and $\text{rank}(\boldsymbol{L}^k)$ denotes the rank of matrix $\boldsymbol{L}^k$. The first term of the objective function attempts to minimize the measured distance between similar samples, while the second term along with the first set of soft constraints (due to the presence of slack variables ξ^k_{mn}) encourage distances between pairs of dissimilar samples to be larger than 1. Evidently, $C > 0$ controls the penalty of violating the previous desiteratum and can be chosen via a validation procedure. Finally, the last term penalizes large ranks of the linear transformations $\boldsymbol{L}^k$. Therefore, the regularization parameter $\lambda \geq 0$, in essence, controls the dimensionality of the feature space.

Problem (2) can be somewhat reformulated by first eliminating the slack variables. Let $[\cdot]_+ : \mathbb{R} \to \mathbb{R}_+$ be the hinge function defined as $[u]_+ \triangleq \max\{u, 0\}$ for all $u \in \mathbb{R}$. It is straightforward to show that $\xi^k_{mn} = \left[1 - \left\| \boldsymbol{L}^k \Delta \boldsymbol{x}_{mn} \right\|_2^2\right]_+$, which can be substituted back into the objective function. Next, we note that $\text{rank}(\boldsymbol{L}^k)$ is a non-convex function w.r.t. $\boldsymbol{L}^k$ and is, therefore, hard to optimize. Following the approaches of [13] and [14], we replace $\text{rank}(\boldsymbol{L}^k)$ with its convex envelope, *i.e.*, the nuclear norm $\boldsymbol{L}^k$, which is defined as the sum of $\boldsymbol{L}^k$'s singular values. These considerations lead to the following problem:

$$\begin{aligned} \underset{\boldsymbol{L}^k, \boldsymbol{g}^k \in \Omega_g}{min} \quad & \sum_k \sum_{m,n} s_{mn} \left\| \boldsymbol{L}^k \Delta \boldsymbol{x}_{mn} \right\|_2^2 g_n^k g_m^k + \\ & + C(1 - s_{mn}) \left[1 - \left\| \boldsymbol{L}^k \Delta \boldsymbol{x}_{mn} \right\|_2^2\right]_+ + \lambda \sum_k \left\| \boldsymbol{L}^k \right\|_* \end{aligned} \tag{3}$$

where $\|\cdot\|_*$ denotes the nuclear norm; in specific, $\left\| \boldsymbol{L}^k \right\|_* \triangleq \sum_{s=1}^P \sigma_s(\boldsymbol{L}^k)$, where σ_s is a singular value of $\boldsymbol{L}^k$.

3 Algorithm

Problem (3) reflects a minimization over two sets of variables. When the $\boldsymbol{g}^k$'s are considered fixed, the problem is non-convex w.r.t. $\boldsymbol{L}^k$, since the second term in Eq. (3) is the combination of a convex function (hinge function) and a non-monotonic function, $1 - \left\|\boldsymbol{L}^k \Delta \boldsymbol{x}_{mn}\right\|_2^2$, w.r.t. $\boldsymbol{L}^k$. On the other hand, if the $\boldsymbol{L}^k$'s are considered fixed, the problem is also non-convex w.r.t $\boldsymbol{g}^k$, since the similarity matrix $\boldsymbol{S}$ is almost always indefinite as it will be argued in the sequel. This implies that the objective function may have multiple minima. Therefore, an iterative procedure seeking to minimize it may have to be started multiple times with different initial estimates of the unknown parameters in order to find its global minimum. In what follows, we discuss a two-step, block-coordinate descent algorithm that is able to perform the minimization in question.

3.1 Two-Step Algorithm

For the first step, we fix $\boldsymbol{g}^k$ and try to solve for each $\boldsymbol{L}^k$. In this case, Problem (3) becomes an unconstrained minimization problem. We observe that the objective function is of the form $f(\boldsymbol{w}) + r(\boldsymbol{w})$, where $\boldsymbol{w}$ is the parameter we are trying to minimize over, $f(\boldsymbol{w})$ is the hinge loss function, which is non-differentiable, and $r(\boldsymbol{w})$ is a non-smooth, convex regularization term. If $f(\boldsymbol{w})$ were smooth, one could employ a proximal gradient method to find a minimum. As this is clearly not the case with the objective function at hand, in our work we resort to using a Proximal Subgradient Descent (PSD) method in a similar fashion to what has been done in [15] and [16]. Moreover, our approach is a special case of [17], based on which we show that our PSD steps converge (see Section 3.2).

Correspondingly, for the second step we assume the $\boldsymbol{L}^k$'s to be fixed and minimize w.r.t. each $\boldsymbol{g}^k$ vector. Consider a matrix $\bar{\boldsymbol{S}}^k$ associated to the k^{th} metric, whose (m, n) element is defined as:

$$\bar{s}_{mn}^k \triangleq s_{mn} \left\|\boldsymbol{L}^k \Delta \boldsymbol{x}_{mn}\right\|_2^2, \quad m, n \in \mathbb{N}_N \tag{4}$$

Then Problem (3) becomes:

$$\underset{\boldsymbol{g}^k \in \Omega_g}{min} \sum_k (\boldsymbol{g}^k)^T \bar{\boldsymbol{S}}^k \boldsymbol{g}^k \tag{5}$$

Let $\boldsymbol{g} \in \mathbb{R}^{KN}$ be the vector that results from concatenating all individual $\boldsymbol{g}^k$ vectors into a single vector and define the matrix

$$\tilde{\boldsymbol{S}} \triangleq \begin{bmatrix} \bar{\boldsymbol{S}}^1 & 0 & \dots & 0 \\ 0 & \bar{\boldsymbol{S}}^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & \bar{\boldsymbol{S}}^K \end{bmatrix} \in \mathbb{R}^{KN \times KN} \tag{6}$$

Based on the previous definitions, the cost function becomes $\boldsymbol{g}^T\tilde{\boldsymbol{S}}\boldsymbol{g}$ and $\boldsymbol{g}$'s constraint set becomes $\Omega_g = \left\{\boldsymbol{g} \in [0,1]^{KN} : \boldsymbol{g} \succeq \boldsymbol{0},\ \boldsymbol{B}\boldsymbol{g} = \boldsymbol{1}\right\}$, where $\boldsymbol{B} \triangleq \boldsymbol{1}^T \otimes \boldsymbol{I}_N$, $\otimes$ denotes the Kronecker product and $\boldsymbol{I}_N$ is the $N \times N$ identity matrix. Hence, the minimization problem for the second step can be re-expressed as:

$$\underset{\boldsymbol{g}\in\boldsymbol{\Omega}_{\boldsymbol{g}}}{min}\ \ \boldsymbol{g}^T\tilde{\boldsymbol{S}}\boldsymbol{g} \tag{7}$$

Problem (7) is non-convex, since $\tilde{\boldsymbol{S}}$ is almost always indefinite. This stems from the fact that $\tilde{\boldsymbol{S}}$ is a block diagonal matrix, whose blocks are Euclidean Distance Matrices (EDMs). It is known that EDMs feature exactly one positive eigenvalue (unless all of them equal 0). Since each EDM is a hollow matrix, its trace equals 0. This, in turn, implies that its remaining eigenvalues must be negative [18]. Hence, $\tilde{\boldsymbol{S}}$ will feature negative eigenvalues.

In order to obtain a minimizer of Problem (7), we employ a Majorization Minimization (MM) approach [19], which first requires identifying a function of $\boldsymbol{g}$ that majorizes the objective function at hand. Let $\mu \triangleq -\lambda_{max}(\tilde{\boldsymbol{S}})$, where $\lambda_{max}(\tilde{\boldsymbol{S}})$ is the largest eigenvalue of $\tilde{\boldsymbol{S}}$. As the latter matrix is indefinite, $\lambda_{max}(\tilde{\boldsymbol{S}}) > 0$. Then, $\boldsymbol{H} \triangleq \tilde{\boldsymbol{S}} + \mu\boldsymbol{I}$ is negative semi-definite. Let $q(\boldsymbol{g}) \triangleq \boldsymbol{g}^T\tilde{\boldsymbol{S}}\boldsymbol{g}$ be the cost function in Eq. (7). Since $(\boldsymbol{g}-\boldsymbol{g}')^T\boldsymbol{H}(\boldsymbol{g}-\boldsymbol{g}') \leq 0$ for any $\boldsymbol{g}$ and $\boldsymbol{g}'$, we have that $q(\boldsymbol{g}) < -\boldsymbol{g}'^T\boldsymbol{H}\boldsymbol{g}' + 2\boldsymbol{g}'^T\boldsymbol{H}\boldsymbol{g} - \mu\left\|\boldsymbol{g}\right\|_2^2$ for all $\boldsymbol{g} \neq \boldsymbol{g}'$ and equality, only if $\boldsymbol{g} = \boldsymbol{g}'$. The right hand side of the aforementioned inequality constitutes q's majorizing function, which we will denote as $q(\boldsymbol{g}|\boldsymbol{g}')$. The majorizing function is used to iteratively optimize $\boldsymbol{g}$ based on the current estimate $\boldsymbol{g}'$. So we have the following minimization problem, which is convex w.r.t $\boldsymbol{g}$:

$$\underset{\boldsymbol{g}\in\Omega_g}{min}\ \ 2\boldsymbol{g}'^T\boldsymbol{H}\boldsymbol{g} - \mu\left\|\boldsymbol{g}\right\|_2^2 \tag{8}$$

This problem is readily solvable, as the next theorem implies.

Theorem 1. *Let $\boldsymbol{g}, \boldsymbol{d} \in \mathbb{R}^{KN}$, $\boldsymbol{B} \triangleq \boldsymbol{1}^T \otimes \boldsymbol{I}_N \in \mathbb{R}^{N\times KN}$ and $c > 0$. The unique minimizer $\boldsymbol{g}^*$ of*

$$\begin{aligned} \underset{\boldsymbol{g}}{min}\ \ & \frac{c}{2}\left\|\boldsymbol{g}\right\|_2^2 + \boldsymbol{d}^T\boldsymbol{g} \\ s.t.\ \ & \boldsymbol{B}\boldsymbol{g} = \boldsymbol{1},\ \boldsymbol{g} \succeq \boldsymbol{0} \end{aligned} \tag{9}$$

has the form

$$g_i^* = \frac{1}{c}\left[(\boldsymbol{B}^T\boldsymbol{\alpha})_i - \boldsymbol{d}_i\right]_+, \quad i \in \mathbb{N}_{KN} \tag{10}$$

where g_i is the i^{th} element of $\boldsymbol{g}$ and $\boldsymbol{\alpha} \in \mathbb{R}^N$ is the Lagrange multiplier vector associated to the equality constraint.

Proof. The Lagrangian of Problem (9) is expressed as:

$$L(\boldsymbol{g}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{c}{2}\boldsymbol{g}^T\boldsymbol{g} + \boldsymbol{d}^T\boldsymbol{g} + \boldsymbol{\alpha}^T(\boldsymbol{1} - \boldsymbol{B}\boldsymbol{g}) - \boldsymbol{\beta}^T\boldsymbol{g} \tag{11}$$

Algorithm 1. Minimization of Problem (3)

Input: Data $\boldsymbol{X} \in R^{D\times N}$, number of metrics K
Output: $\boldsymbol{L}^k, \boldsymbol{g}^k \quad k \in \mathbb{N}_K$
01. Initialize $\boldsymbol{L}^k, \boldsymbol{g}^k$ for all $k \in \mathbb{N}_K$
02. **While** not converged **Do**
03. Step 1: Use a PSD method to solve Problem (3) for each $\boldsymbol{L}^k$
04. Step 2:
05. $\tilde{\boldsymbol{S}} \leftarrow Eq.$ (6)
06. $\mu \leftarrow -\lambda_{max}(\tilde{\boldsymbol{S}})$
07. $\boldsymbol{H} \leftarrow \tilde{\boldsymbol{S}} + \mu \boldsymbol{I}$
08. **While** not converged **Do**
09. Apply binary search to obtain each $\boldsymbol{g}^k$ using Eq. (10)
10. **End While**
11. **End While**

where $\boldsymbol{\alpha} \in \mathbb{R}^N$ and $\boldsymbol{\beta} \in \mathbb{R}^{KN}$ with $\boldsymbol{\beta} \succeq \mathbf{0}$ are Lagrange multiplier vectors. If we set the partial derivative of $L(\boldsymbol{g}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ with respect to $\boldsymbol{g}$ to $\mathbf{0}$, we readily obtain that

$$g_i = \frac{1}{c}\left((\boldsymbol{B}^T\boldsymbol{\alpha})_i + \beta_i - d_i\right), \quad i \in \mathbb{N}_{KN} \tag{12}$$

Let $\gamma_i \triangleq (\boldsymbol{B}^T\boldsymbol{\alpha})_i - d_i$. Combining Eq. (12) with the complementary slackness condition $\beta_i g_i = 0$, one obtains that, if $\gamma_i \leq 0$, then $\beta_i = -\gamma_i$ and $g_i = 0$, while, when $\gamma_i > 0$, then $\beta_i = 0$ and, evidently, $g_i = \frac{1}{c}\gamma_i$. These two observations can be summarized into $g_i = \frac{1}{c}[\gamma_i]_+$, which completes the proof. □

In order to exploit the result of Theorem 1 for obtaining a concrete solution to Problem (8), we ought to point out that the (unknown) optimal values of the Lagrange multipliers α_i can be found via binary search, so they satisfy the equality constraint $\boldsymbol{B}\boldsymbol{g} = \mathbf{1}$.

In conclusion, the entire algorithm for solving Problem (3) can be recapitulated as follows: For step 1, the $\boldsymbol{g}^k$ vectors are assumed fixed and a PSD is being employed to minimize the cost function of Eq. (3) w.r.t. each weight matrix $\boldsymbol{L}^k$. For step 2, all $\boldsymbol{L}^k$'s are held fixed to the values obtained after completion of the previous step and the solution offered by Theorem 1 along with binary searches for the α_i's are used to compute the optimal $\boldsymbol{g}_k$'s by iteratively solving Problem (8) via a MM scheme. Note that these two main steps are repeated until convergence is established; the whole process is depicted in Algorithm 1.

3.2 Analysis

In this subsection, we investigate the convergence of our proposed algorithm. Suppose that a PSD method is employed to minimize the function $f(\boldsymbol{w}) + r(\boldsymbol{w})$, where both f and r are non-differentiable. Denote ∂f as the subgradient of f and

define $\|\partial f(\boldsymbol{w})\| \triangleq \sup_{\boldsymbol{g}\in\partial f(\boldsymbol{w})} \|\boldsymbol{g}\|$; the corresponding quantities for r are similarly defined. Like in [20] and [21], we assume that the subgradients are bounded, *i.e.*:

$$\|\partial f(\boldsymbol{w})\|^2 \leq Af(\boldsymbol{w}) + G^2, \quad \|\partial r(\boldsymbol{w})\|^2 \leq Ar(\boldsymbol{w}) + G^2 \tag{13}$$

where A and G are scalars. Let $\boldsymbol{w}^*$ be a minimizer of $f(\boldsymbol{w}) + r(\boldsymbol{w})$. Then we have the following lemma for the problem under consideration.

Lemma 2. *Suppose that a PSD method is employed to solve* $\min_{\boldsymbol{w}} f(\boldsymbol{w})+r(\boldsymbol{w})$. *Assume that 1) f and r are lower-bounded; 2) the norms of any subgradients ∂f and ∂r are bounded as in Eq. (13); 3) $\|\boldsymbol{w}^*\| \leq D$ for some $D > 0$; 4) $r(\mathbf{0}) = 0$. Let $\eta_t \triangleq \frac{D}{\sqrt{8T}G}$, where T is the number of iterations of the PSD algorithm. Then, for a constant $c \leq 4$, such that $(1 - cA\frac{D}{\sqrt{8T}D}) > 0$, and initial estimate of the solution $\boldsymbol{w}_1 = \mathbf{0}$, we have:*

$$\min_{t\in\{1\ldots T\}} f(\boldsymbol{w}_t) + r(\boldsymbol{w}_t) \leq \frac{1}{T}\sum_{t=1}^{T} f(\boldsymbol{w}_t) + r(\boldsymbol{w}_t) \leq$$
$$\leq \frac{4\sqrt{2}DG}{\sqrt{T}(1 - \frac{cAD}{G\sqrt{8T}})} + \frac{f(\boldsymbol{w}^*) + r(\boldsymbol{w}^*)}{1 - \frac{cAD}{G\sqrt{8T}}} \tag{14}$$

The proof of Lemma 2 is straightforward as it is based on [17] and, therefore, is omitted here. Lemma 2 implies that, as T grows, the PSD iterates approach $\boldsymbol{w}^*$.

Theorem 3. *Algorithm 1 yields a convergent, non-increasing sequence of cost function values relevant to Problem (3). Furthermore, the set of fixed points of the iterative map embodied by Algorithm 1 includes the KKT points of Problem (3).*

Proof. We first prove that each of the two steps in our algorithm decreases the objective function value. This is true for the first step, according to Lemma 2. For the second step, since a MM algorithm is used, we have the following relationships

$$q(\boldsymbol{g}^*) = q(\boldsymbol{g}^*|\boldsymbol{g}^*) \leq q(\boldsymbol{g}^*|\boldsymbol{g}') \leq q(\boldsymbol{g}'|\boldsymbol{g}') = q(\boldsymbol{g}') \tag{15}$$

This implies that the second step always decreases the objective function value. Since the objective function is lower bounded, our algorithm converges.

Next, we prove that the set of fixed points of the proposed algorithm includes the KKT points of Problem (3). Towards this purpose, suppose the algorithm has converged to a KKT point $\left\{\boldsymbol{L}^{k*}, \boldsymbol{g}^{k*}\right\}_{k\in\mathbb{N}_K}$; then, it suffices to show that this point is also a fixed point of the algorithm's iterative map. For notational brevity, let $f_0(\boldsymbol{L}^k, \boldsymbol{g}^k)$, $f_1(\boldsymbol{g}^k)$ and $h_1(\boldsymbol{g}^k)$ be the cost function, inequality constraint and equality constraint of Problem (3) respectively. By definition, the KKT point will satisfy

$$\begin{aligned} &\mathbf{0} \in \partial_{\boldsymbol{L}^k} f_0(\boldsymbol{L}^{k*}, \boldsymbol{g}^{k*}) + \bigtriangledown_{\boldsymbol{g}^k} f_0(\boldsymbol{L}^{k*}, \boldsymbol{g}^{k*}) \\ &- (\boldsymbol{\beta}^k)^T \bigtriangledown_{\boldsymbol{g}^k} f_1(\boldsymbol{g}^{k*}) + \boldsymbol{\alpha}^T \bigtriangledown_{\boldsymbol{g}^k} h_1(\boldsymbol{g}^{k*}) \quad k \in \mathbb{N}_K \end{aligned} \tag{16}$$

In relation to Problem (7), which step 2 tries to solve, the KKT point will satisfy the following equality (gradient of the problem's Lagrangian set to $\mathbf{0}$):

$$2\tilde{\boldsymbol{S}}\boldsymbol{g}^* - \boldsymbol{\beta} - \boldsymbol{B}^T\boldsymbol{\alpha} = \mathbf{0} \tag{17}$$

Problem (8) can be solved based on Eq. (12) of Theorem 1; in specific, we obtain that

$$\boldsymbol{g} = -\frac{1}{2\boldsymbol{\mu}}(\boldsymbol{B}^T\boldsymbol{\alpha} + \boldsymbol{\beta} - 2\boldsymbol{H}\boldsymbol{g}^*) \tag{18}$$

Substituting Eq. (17) and $\boldsymbol{H} = \tilde{\boldsymbol{S}} + \mu\boldsymbol{I}$ into Eq. (18), one immediately obtains that

$$\boldsymbol{g} = -\frac{1}{2\boldsymbol{\mu}}(\boldsymbol{B}^T\boldsymbol{\alpha} + \boldsymbol{\beta} - 2\boldsymbol{H}\boldsymbol{g}^*) = -\frac{1}{2\boldsymbol{\mu}}(2\tilde{\boldsymbol{S}}\boldsymbol{g}^* - 2\tilde{\boldsymbol{S}}\boldsymbol{g}^* - 2\boldsymbol{\mu}\boldsymbol{g}^*) = \boldsymbol{g}^* \tag{19}$$

In other words, step 2 will not update the solution. Now, if we substitute Eq. (17) back into Eq. (16), we obtain $\mathbf{0} \in \partial_{\boldsymbol{L}^k} f_0(\boldsymbol{L}^{k*}, \boldsymbol{g}^{k*})$ for all k, which is the optimality condition for the subgradient method; the PSD step (step 1 of our algorithm) will also not update the solution. Thus, a KKT point of Problem (3) is a fixed point of our algorithm. □

Table 1. Details of benchmark data sets. For the Letter and Pendigits datasets, only 4 and 5 classes were considered respectively.

	#D	#CLASSES	#TRAIN	#VALIDATION	#TEST
ROBOT	4	4	240	240	4976
LETTER A-D	16	4	200	400	2496
PENDIGITS 1-5	16	5	200	1800	3541
WINEQUALITY	12	2	150	150	6197
TELESCOPE	10	2	300	300	11400
IMGSEG	18	7	210	210	1890
TWONORM	20	2	250	250	6900
RINGNORM	20	2	250	250	6900
IONOSPHERE	34	2	80	50	221

4 Experiments

In this section, we performed experiments on 9 datasets, namely, Robot Navigation, Letter Recognition, Pendigits, Wine Quality, Gamma Telescope, Ionosphere datasets from the *UCI machine learning repository*[1], and Image Segmentation, Two Norm, Ring Norm datasets from the *Delve Dataset Collection*[2]. Some characteristics of these datasets are summarized in Table 1. We first explored how

[1] `http://archive.ics.uci.edu/ml/datasets.html`
[2] `http://www.cs.toronto.edu/~delve/data/datasets.html`

the performance of R^{2}LML[1] varies with respect to the number of local metrics. Then, we compared R^{2}LML with other global or local Metric Learning algorithms, including ITML, LMNN, LMNN-MM, GLML and PLML.

The computation of the distances between some test sample $\boldsymbol{x}$ and the training samples $\boldsymbol{x}_n$ according to our formulation requires the value of $\boldsymbol{g}$ corresponding to $\boldsymbol{x}$. One option to assign a value to $\boldsymbol{g}$ would be to utilize transductive learning. However, as such an approach could prove computationally expensive, we opted instead to assign $\boldsymbol{g}$ the value of the corresponding vector associated to $\boldsymbol{x}$'s nearest (in terms of Euclidean distance) training sample as was done in [12].

4.1 Number of Local Metrics

In this subsection, we show how the performance of R^{2}LML varies with respect to the number of local metrics K. In [9], the authors set K equal to the number of classes for each dataset, which might not necessarily be the optimal choice. In our experiments, we let K vary from 1 to 7. This range covers the maximum number of classes in the datasets that are considered in our experiments. As we will show, the optimal K is not always the same as the number of classes.

Besides K, we held the remaining parameters (refer to Eq. (2)) fixed: the penalty parameter C was set to 1 and the nuclear norm regularization parameter λ to 0.1. Moreover, we terminated our algorithm, if it reached 10 epochs or when the difference of cost function values between two consecutive iterations was less than 10^{-4}. In each epoch, the PSD inner loop ran for 500 iterations. The PSD step length was fixed to 10^{-5} for the Robot and Ionosphere datasets, to 10^{-6} for the Letter A-D, Two norm and Ring Norm datasets, to 10^{-8} for the Pendigits 1-5, Wine Quality and Image Segmentation datasets and to 10^{-9} for the Gamma Telescope dataset. The MM loop was terminated, if the number of iterations reached 3000 or when difference of cost function values between two consecutive iterations was less than 10^{-3}. The relation between number of local metrics and the classification accuracy for each dataset is reported in Figure 2.

Several observations can be made based on Figure 2. First of all, our method used as a local metric learning method (when $K \geq 2$) performs much better than when used with a single global metric (when $K = 1$) for all datasets except the Ring Norm dataset. For the latter dataset, the classification performance deteriorates with increasing K. Secondly, one cannot discern a deterministic relationship between the classification accuracy and the number of local metrics utilized that is suitable for all datasets. For example, for the Robot dataset, the classification accuracy is almost monotonically increasing with respect to K. For the remaining datasets, the optimal K varies in a non-apparent fashion with respect to their number of classes. For example, in the case of the Ionosphere dataset (2-class problem), $K = 3, 6, 7$ yield the best generalization results. All these observations suggest that validation over K is needed to select the best performing model.

[1] `https://github.com/yinjiehuang/R2LML/archive/master.zip`

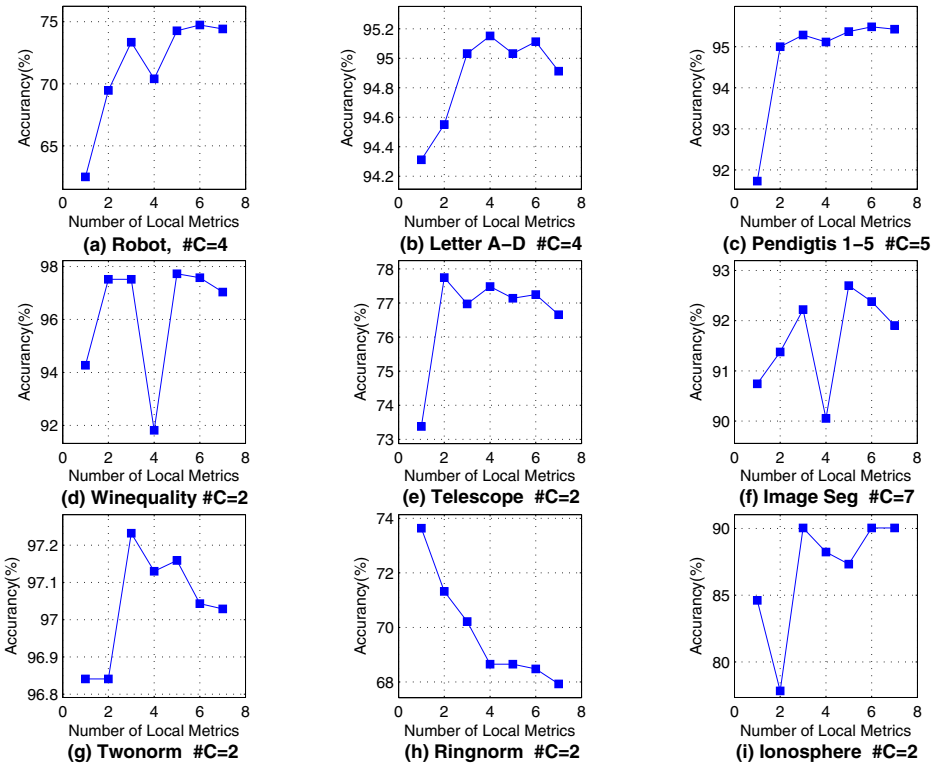

Fig. 2. R^{2}LML classification accuracy results on the 9 benchmark datasets for varying number K of local metrics. $\#C$ indicates the number of classes of each dataset.

4.2 Comparisons

We compared R^{2}LML with several other metric learning algorithms, including Euclidean metric KNN, ITML [6], LMNN [5], LMNN-MM [9], GLML [11] and PLML [12]. Both ITML and LMNN learn a global metric, while LMNN-MM, GLML and PLML are local metric learning algorithms. After the metrics are learned, the KNN classifier is utilized for classification with k (number of nearest neighbors) set to 5.

For our experiments we used LMNN, LMNN-MM[1], ITML[2] and PLML[3] implementations that we found available online. For ITML, a good value of γ is found via cross-validation. Also, for LMNN and LMNN-MM, the number of attracting neighbors during training is set to 1. Additionally, for LMNN, at most 500 iterations were performed and 30% of training data were used as a validation

[1] `http://www.cse.wustl.edu/~kilian/code/code.html`
[2] `http://www.cs.utexas.edu/~pjain/itml/`
[3] `http://cui.unige.ch/~wangjun/papers/PLML.zip`

Table 2. Percent accuracy results of 7 algorithms on 9 benchmark datasets. For each dataset, the statistically best and comparable results for a family-wise significance level of 0.05 are highlighted in boldface. All algorithms are ranked from best to worst; algorithms share the same rank, if their performance is statistically comparable.

	Euclidean	ITML	LMNN	LMNN-MM	GLML	PLML	R^2LML
Robot	65.31^{2nd}	65.86^{2nd}	66.10^{2nd}	66.10^{2nd}	62.28^{3rd}	61.03^{3rd}	$\mathbf{74.16}^{1st}$
Letter A-D	88.82^{2nd}	$\mathbf{93.39}^{1st}$	$\mathbf{93.79}^{1st}$	$\mathbf{93.83}^{1st}$	89.30^{2nd}	$\mathbf{94.43}^{1st}$	$\mathbf{95.07}^{1st}$
Pendigits 1-5	88.31^{4th}	93.17^{2nd}	91.19^{3rd}	91.27^{3rd}	88.37^{4th}	$\mathbf{95.88}^{1st}$	$\mathbf{95.43}^{1st}$
Winequality	86.12^{7th}	96.11^{3rd}	94.43^{4th}	93.38^{5th}	91.79^{6th}	$\mathbf{98.55}^{1st}$	97.53^{2nd}
Telescope	70.31^{3rd}	71.42^{2nd}	72.16^{2nd}	71.45^{2nd}	70.31^{3rd}	$\mathbf{77.52}^{1st}$	$\mathbf{77.97}^{1st}$
ImgSeg	80.05^{4th}	90.21^{2nd}	90.74^{2nd}	89.42^{2nd}	87.30^{3rd}	90.48^{2nd}	$\mathbf{92.59}^{1st}$
Twonorm	96.54^{2nd}	$\mathbf{96.78}^{1st}$	96.32^{2nd}	96.30^{2nd}	96.52^{2nd}	$\mathbf{97.32}^{1st}$	$\mathbf{97.23}^{1st}$
Ringnorm	55.84^{7th}	77.35^{2nd}	59.36^{6th}	59.75^{5th}	$\mathbf{97.09}^{1st}$	75.68^{3rd}	73.73^{4th}
Ionosphere	75.57^{3rd}	$\mathbf{86.43}^{1st}$	82.35^{2nd}	82.35^{2nd}	71.95^{3rd}	78.73^{3rd}	$\mathbf{90.50}^{1st}$

set. The maximum number of iterations for LMNN-MM was set to 50 and a step size of 10^{-7} was employed. For GLML, we chose γ by maximizing performance over a validation set. Finally, the PLML hyperparameter values were chosen as in [12], while α_1 was chosen via cross-validation. With respect to R^2LML, for each dataset we used K's optimal value as established in the previous series of experiments, while the regularization parameter λ was chosen via a validation procedure over the set $\{0.01, 0.1, 1, 10, 100\}$. The remaining parameter settings of our method were the same as the ones used in the previous experiments.

For pair-wise model comparisons, we employed McNemar's test. Since there are 7 algorithms to be compared, we used Holm's step-down procedure as a multiple hypothesis testing method to control the Family-Wise Error Rate (FWER) [22] of the resulting pair-wise McNemar's tests. The experimental results for a family-wise significance level of 0.05 are reported in Table 2.

It is observed that R^2LML achieves the best performance on 7 out of the 9 datasets, while GLML, ITML and PLML outperform our model on the Ring Norm dataset. GLML's surprisingly good result for this particular dataset is probably because GLML assumes a Gaussian mixture underlying the data generation process and the Ring Norm dataset is a 2-class recognition problem drawn from a mixture of two multivariate normal distributions. Even though not being the best model for this dataset, R^2LML is still highly competitive compared to LMNN, LMNN-MM and Euclidean KNN. Next, PLML performs best in 5 out of 9 datasets, even outperforming R^2LML on the Wine Quality dataset. However, PLML gives poor results on some datasets like Robot or Ionosphere. Also, PLML does not show much improvements over KNN and may even perform worse like for the Robot dataset. Note, that R^2LML is still better for the Image Segmentation, Robot and Ionosphere datasets. Additionally, ITML is ranked first for 3 datasets and even outperforms R^2LML on the Ring Norm dataset. Often, ITML ranks at least 2^{nd} and seems to be suitable for low dimensional datasets. However, R^2LML still performs better than ITML for 5 out of the 9 datasets. Finally,

GLML rarely performs well; according to Table 2, GLML only achieves 3^{rd} or 4^{th} ranks for 6 out of the 9 datasets.

Another general observation that can be made is the following: employing metric learning is almost always a good choice, since the classification accuracy of utilizing a Euclidean metric is almost always the lowest among all the 7 methods we considered. Interestingly, LMNN-MM, even though being a local metric learning algorithm, does not show any performance advantages over LMNN (a global metric method); for some datasets, it even obtained lower classification accuracy than LMNN. It is possible that fixing the number of local metrics to the number of classes present in the dataset curtails LMNN-MM's performance. According to the obtained results, R^{2}LML yields much better performance for all datasets compared to LMNN-MM. This consistent performance advantage may not only be attributed to the fact that K was selected via a validation procedure, since, for cases where the optimal K equaled the number of classes (*e.g.* Letter A-D dataset), R^{2}LML still outperformed LMNN-MM.

5 Conclusions

In this paper, we proposed a new local metric learning model, namely Reduced-Rank Local Metric Learning (R^{2}LML). It learns K Mahalanobis-based local metrics that are conically combined, such that similar points are closer to each other, while the separation between dissimilar ones is encouraged to increase. Additionally, a nuclear norm regularizer is adopted to obtain low-rank weight matrices for calculating metrics. In order to solve our proposed formulation, a two-step algorithm is showcased, which iteratively solves two sub-problems in an alternating fashion; the first sub-problem is minimized via a Proximal Subgradient Descent (PSD) approach, while the second one via a Majorization Minimization (MM) procedure. Moreover, we have demonstrated that our algorithm converges and that its fixed points include the Karush-Kuhn-Tucker (KKT) points of our proposed formulation.

In order to show the merits of R^{2}LML, we performed a series of experiments involving 9 benchmark classification problems. First, we varied the number of local metrics K and discussed the influence of K on classification accuracy. We concluded that there is no obvious relation between K and the classification accuracy. Furthermore, the obtained optimal K does not necessarily equal the number of classes of the dataset under consideration. Finally, in a second set of experiments, we compared R^{2}LML to several other metric learning algorithms and demonstrated that our proposed method is highly competitive.

Acknowledgments. Y. Huang acknowledges partial support from a UCF Graduate College Presidential Fellowship and National Science Foundation (NSF) grant No. 1200566. C. Li acknowledges partial support from NSF grants No. 0806931 and No. 0963146. Furthermore, M. Georgiopoulos acknowledges partial support from NSF grants No. 1161228 and No. 0525429, while G. C. Anagnostopoulos acknowledges partial support from NSF grant No. 1263011. Note that

any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF. Finally, the authors would like to thank the 3 anonymous reviewers of this manuscript for their helpful comments.

References

1. Xing, E.P., Ng, A.Y., Jordan, M.I., Russell, S.: Distance metric learning with application to clustering with side-information. In: Neural Information Processing Systems Foundation (NIPS), pp. 505–512. MIT Press (2002)
2. Shalev-Shwartz, S., Singer, Y., Ng, A.Y.: Online and batch learning of pseudo-metrics. In: International Conference on Machine Learning (ICML). ACM (2004)
3. Chopra, S., Hadsell, R., Lecun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Computer Vision and Pattern Recognition (CVPR), pp. 539–546. IEEE Press (2005)
4. Goldberger, J., Roweis, S., Hinton, G., Salakhutdinov, R.: Neighbourhood components analysis. In: Neural Information Processing Systems Foundation (NIPS), pp. 513–520. MIT Press (2004)
5. Weinberger, K.Q., Blitzer, J., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. In: Neural Information Processing Systems Foundation (NIPS). MIT Press (2006)
6. Davis, J.V., Kulis, B., Jain, P., Sra, S., Dhillon, I.S.: Information-theoretic metric learning. In: International Conference on Machine Learning (ICML), pp. 209–216. ACM (2007)
7. Yang, L., Jin, R., Sukthankar, R., Liu, Y.: An efficient algorithm for local distance metric learning. In: AAAI Conference on Artificial Intelligence (AAAI). AAAI Press (2006)
8. Hastie, T., Tibshirani, R.: Discriminant adaptive nearest neighbor classification. IEEE Transactions on Pattern Analysis and Machine Intelligence 18(6), 607–616 (1996)
9. Weinberger, K., Saul, L.: Fast solvers and efficient implementations for distance metric learning. In: International Conference on Machine Learning (ICML), pp. 1160–1167. ACM (2008)
10. Bilenko, M., Basu, S., Mooney, R.J.: Integrating constraints and metric learning in semi-supervised clustering. In: International Conference on Machine Learning (ICML), pp. 81–88. ACM (2004)
11. Noh, Y.K., Zhang, B.T., Lee, D.D.: Generative local metric learning for nearest neighbor classification. In: Neural Information Processing Systems Foundation (NIPS). MIT Press (2010)
12. Wang, J., Kalousis, A., Woznica, A.: Parametric local metric learning for nearest neighbor classification. In: Neural Information Processing Systems Foundation (NIPS), pp. 1610–1618. MIT Press (2012)
13. Candès, E.J., Tao, T.: The power of convex relaxation: Near-optimal matrix completion. CoRR abs/0903.1476 (2009)
14. Candès, E.J., Recht, B.: Exact matrix completion via convex optimization. CoRR abs/0805.4471 (2008)
15. Rakotomamonjy, A., Flamary, R., Gasso, G., Canu, S.: lp-lq penalty for sparse linear and sparse multiple kernel multi-task learning. IEEE Transactions on Neural Networks 22, 1307–1320 (2011)

16. Chen, X., Pan, W., Kwok, J.T., Carbonell, J.G.: Accelerated gradient method for multi-task sparse learning problem. In: International Conference on Data Mining (ICDM), pp. 746–751. IEEE Computer Society (2009)
17. Duchi, J., Singer, Y.: Efficient online and batch learning using forward backward splitting. Journal of Machine Learning Research (JMLR) 10, 2899–2934 (2009)
18. Balaji, R., Bapat, R.: On euclidean distance matrices. Linear Algebra and its Applications 424(1), 108–117 (2007)
19. Hunter, D.R., Lange, K.: A tutorial on mm algorithms. The American Statistician 58(1) (2004)
20. Langford, J., Li, L., Zhang, T.: Sparse online learning via truncated gradient. Journal of Machine Learning Research (JMLR) 10, 777–801 (2009)
21. Shalev-Shwartz, S., Tewari, A.: Stochastic methods for l1-regularized loss minimization. Journal of Machine Learning Research (JMLR) 12, 1865–1892 (2011)
22. Hochberg, Y., Tamhane, A.C.: Multiple comparison procedures. John Wiley & Sons, Inc., New York (1987)