Transfer Metric Learning by Learning Task Relationships

Yu Zhang and Dit-Yan Yeung
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
{zhangyu, dyyeung}@cse.ust.hk

ABSTRACT

Distance metric learning plays a very crucial role in many data mining algorithms because the performance of an algorithm relies heavily on choosing a good metric. However, the labeled data available in many applications is scarce and hence the metrics learned are often unsatisfactory. In this paper, we consider a transfer learning setting in which some related source tasks with labeled data are available to help the learning of the target task. We first propose a convex formulation for multi-task metric learning by modeling the task relationships in the form of a task covariance matrix. Then we regard transfer learning as a special case of multi-task learning and adapt the formulation of multi-task metric learning to the transfer learning setting for our method, called transfer metric learning (TML). In TML, we learn the metric and the task covariances between the source tasks and the target task under a unified convex formulation. To solve the convex optimization problem, we use an alternating method in which each subproblem has an efficient solution. Experimental results on some commonly used transfer learning applications demonstrate the effectiveness of our method.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; H.2.8 [**Database Management**]: Database Applications—*Data mining*

General Terms

Algorithms

Keywords

Metric Learning, Transfer Learning, Multi-Task Learning

1. INTRODUCTION

Many data mining algorithms, e.g., k-means clustering algorithm and k-nearest neighbor classifier, work by relying on a distance metric. In order to deliver satisfactory results, finding a good distance metric for the problem at hand often plays a very crucial role. As such, metric learning [25] has received much attention in the research community [12, 25, 5, 22, 8, 26, 6, 7, 27, 29, 13]. Many metric learning methods have been proposed. From the perspective of the underlying learning paradigm, these methods can be grouped into three categories, namely, supervised metric learning, unsupervised metric learning, and semi-supervised metric learning. Supervised metric learning learns a metric for some supervised learning task, such as classification, so that data points from the same class are kept close while those from different classes are far apart [12, 22, 8, 7, 29, 13]. It has also been used for regression by exploiting the manifold structure contained in the labeled data [24]. Unsupervised metric learning utilizes some information contained in the data to learn a metric for some unsupervised learning task, such as clustering [6]. Semi-supervised metric learning, which can be viewed as a combination of the supervised and unsupervised paradigms, utilizes both label information from the labeled data and geometric information from the unlabeled data to learn a good metric for classification or clustering. The need for semi-supervised metric learning arises from the fact that the labeled data available in a number of real-life applications is scarce because labeling data is very laborious and costly. With only limited labeled data, the metrics learned are often unsatisfactory. Semi-supervised metric learning tries to exploit additional information from the unlabeled data to alleviate this problem which is known as labeled data deficiency problem here.

The focus of this work is on supervised metric learning for classification applications. However, we consider situations similar to those for semi-supervised metric learning in which there is deficiency in labeled data. While the amount of labeled data available in one learning task is limited, it is not uncommon that there exist other related learning tasks with labeled data available. Unlike semi-supervised learning which exploits unlabeled data, multi-task learning [4, 12, 20] and transfer learning [18] seek to alleviate the labeled data deficiency problem by utilizing some related learning tasks to help improve the learning performance. In some sense, they mimic human learning activities in that people may learn faster when several related tasks are learned simultaneously, e.g., playing different games. In essence, people often apply the knowledge gained from some previous learning tasks to help learn a new task. Even though both multi-task learning and transfer learning utilize information from other related learning tasks, there exist some differences between them in both the problem setting and the objective. In transfer learning, the learning tasks are usually classified into two types: source task and target task. It is assumed that there is enough data in the source tasks but not in the target task. The objective of transfer learning is to utilize the information in the source tasks to help learn the target task with no need for improving the performance of the source tasks. On the other hand, there is no distinction between the tasks in multi-task learning and the objective is to improve the performance of all tasks simultaneously.

Even though there exist differences between multi-task learning and transfer learning, a central issue common to both is to accurately characterize the relationships between tasks. Given the training data for multiple tasks, there are two important aspects that distinguish between different methods for characterizing the task relationships. The first aspect is on *how* to obtain the relationships, either from the model assumption or automatically learned from data. Many multi-task and transfer learning methods make some prior assumptions. For example, the latent data representation is shared by different tasks [4, 1] or the learning models in different tasks have similar model parameters [9, 14]. Obviously, learning the task relationships from data automatically is the more favorable option because the model assumption adopted may be incorrect and, worse still, it is not easy to verify the correctness of the assumption from data. The second aspect is on *what* task relationships can be represented by a method. Generally speaking there are three types of task relationship: positive task correlation, negative task correlation, and task unrelatedness.[1] Positive task correlation is a useful task relationship to characterize because similar tasks are likely to have similar model parameters. For negative task correlation, knowing the model parameters of one task will reduce the search space for the model parameters of a negatively correlated task. As for task unrelatedness, identifying outlier tasks can prevent them from impairing the performance of other tasks since outlier tasks are unrelated to other tasks.

In this paper, we study metric learning under the transfer learning setting in which some source tasks are available in addition to the target task. Based on a method called regularized distance metric learning (RDML) [13], we propose an extension for transfer learning called *transfer metric learning* (TML). Different from conventional transfer learning methods, we first propose a convex formulation for multi-task metric learning by modeling the task relationships in the form of a task covariance matrix which can model positive, negative and zero task correlations. Then we regard transfer learning as a special case of multi-task learning in that the source tasks are equally important and independent, and adapt the formulation of multi-task metric learning to the transfer learning setting for the formulation of TML. In TML, we learn the metric and the task covariances between the source tasks and the target task under a unified convex formulation. As in multi-task metric learning, the task covariance matrix can also model positive, negative and zero task correlations. To solve the convex optimization problem, we use an alternating method in which each subproblem has an efficient solution. Experimental results on some commonly used transfer learning applications demonstrate the effectiveness of our method.

[1]Task unrelatedness corresponds to zero or close to zero task correlation.

The remainder of this paper is organized as follows. We first briefly introduce some background for metric learning and the related work in Section 2. We then present our multi-task metric learning and TML algorithms in Sections 3 and 4, respectively. Section 5 reports experimental results on some transfer learning applications. Finally, some concluding remarks are given in the last section.

2. BACKGROUND AND RELATED WORK

Suppose we are given a labeled training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where the ith data point $\mathbf{x}_i \in \mathbb{R}^d$ and its class label $y_i \in \{1, \ldots, C\}$. In RDML [13], the learning problem is formulated as follows:

$$\begin{aligned} \min_{\mathbf{\Sigma}} \quad & \frac{2}{n(n-1)} \sum_{j<k} g\Big(y_{j,k}\left[1 - \|\mathbf{x}_j - \mathbf{x}_k\|_{\mathbf{\Sigma}}^2\right]\Big) + \frac{\lambda}{2}\|\mathbf{\Sigma}\|_F^2 \\ \text{s.t.} \quad & \mathbf{\Sigma} \succeq \mathbf{0}, \end{aligned} \tag{1}$$

where λ is the regularization parameter which balances the empirical loss and the regularization term, $\|\cdot\|_F$ denotes the Frobenius norm of a matrix, $y_{j,k}$ is equal to 1 when y_j and y_k are identical and -1 otherwise, $\mathbf{\Sigma} \succeq \mathbf{0}$ means that $\mathbf{\Sigma}$ is a positive semidefinite (PSD) matrix, $\|\mathbf{x}_j - \mathbf{x}_k\|_{\mathbf{\Sigma}}^2 = (\mathbf{x}_j - \mathbf{x}_k)^T \mathbf{\Sigma}(\mathbf{x}_j - \mathbf{x}_k)$, $g(z) = \max(0, b - z)$ which is similar to the hinge loss used in the support vector machine (SVM). Here b is a constant, satisfying $0 \leq b \leq 1$, which denotes the classification margin. In [13], b is set to 0.

In [13], an online method is used to learn the optimal $\mathbf{\Sigma}$ and some properties of RDML, such as the generalization error, are studied. Moreover, theoretical analysis shows that RDML is robust against the number of feature dimensions.

To the best of our knowledge, [28] is the only previous work on transfer metric learning. In [28], it is assumed that there exist labeled data points for the target task as well as some prior information from the source tasks in the form of a metric matrix learned from each source task. The authors extended information-theoretic metric learning (ITML) [8] to transfer metric learning by treating the metric matrices learned from the source tasks as prior information to regularize the learning of the target task. The optimization problem for transfer metric learning in [28], which is called L-DML, is formulated as follows:

$$\begin{aligned} \min_{\mathbf{M}} \quad & \sum_{k=1}^{K} \mu_k \mathrm{tr}(\mathbf{M}_k^{-1}\mathbf{M}) - \log|\mathbf{M}| + \eta_s \mathrm{tr}(\mathbf{SM}) - \eta_d \mathrm{tr}(\mathbf{DM}) \\ & +\gamma\|\boldsymbol{\mu}\|_2^2 \\ \text{s.t.} \quad & \mathbf{M} \succeq 0 \\ & \sum_{k=1}^{K} \mu_k = 1, \mu_k \geq 0, \end{aligned} \tag{2}$$

where $\mathrm{tr}(\cdot)$ denotes the trace of a square matrix and $\|\cdot\|_2$ denotes the 2-norm of a vector. Here $\mathbf{S} = \sum_{y_i=y_j}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$, $\mathbf{D} = \sum_{y_i \neq y_j}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$, and $\mathbf{M}_k$ ($k = 1, \ldots, K$) is the available metric matrix for the kth source task. The first and second terms in the objective function of problem (2) are derived from the log-determinant regularization function as used in [8] and μ_k is the weight that reflects the utility of the metric of the kth source task. The third term is to keep the data points in the same class as close as possible and the fourth term is to keep the data points from different classes far apart. The last term is to penalize the complexity of $\boldsymbol{\mu}$. Here $\boldsymbol{\mu}$ plays an important role in this formulation since there may exist outlier tasks in real applications and by learning $\boldsymbol{\mu}$ L-DML can identity them. However, each element of the vector $\boldsymbol{\mu}$ is non-negative and so it cannot model the negative transfer situation [19]. Moreover, the constraint $\sum_{k=1}^{K} \mu_k = 1$ is not very reasonable. Consider a special case in which there is only one source task. Then $\mu_1 = 1$ even if this source task is an outlier task. When there are multiple source tasks and all of them are outlier tasks, we should set all μ_i to zero but then the constraint $\sum_{k=1}^{K} \mu_k = 1$ cannot be satisfied. Furthermore, problem (2) is not convex, making it easy to get trapped in (bad) local minima during the optimization procedure.

There exist some methods for transfer dimensionality reduction [21, 16, 17], where dimensionality reduction can be viewed as a special case of metric learning in that the metric learned is not of full rank. However, transfer dimensionality reduction is different from transfer metric learning and these methods are not applicable here. For example, [21] used a transformation matrix for dimensionality reduction in the source tasks for subspace clustering in the target task and so the target task is an unsupervised learning task. Also, [16, 17] proposed dimensionality reduction methods for domain adaption in which the target task has no labeled data, and so it is different from the setting here where we utilize the metric matrices learned from the source tasks to help the learning of the target task from labeled data.

3. MULTI-TASK METRIC LEARNING

In this section, we propose a multi-task metric learning method which can learn the task relationships between all pairs of tasks.

Suppose we are given m learning tasks $\{T_i\}_{i=1}^{m}$. For the ith task T_i, the training set $\mathcal{D}_i$ consists of n_i data points represented in the form of $(\mathbf{x}_j^i, y_j^i)$, $j = 1, \ldots, n_i$, with $\mathbf{x}_j^i \in \mathbb{R}^d$ and its corresponding class label $y_j^i \in \{1, \ldots, C_i\}$. Here the superscript denotes the task index and the subscript denotes the instance index in each task.

The optimization problem for multi-task metric learning is formulated as follows:

$$\begin{aligned}
\min_{\{\boldsymbol{\Sigma}_i\}, \boldsymbol{\Omega}} \quad & \sum_{i=1}^{m} \frac{2}{n_i(n_i-1)} \sum_{j<k} g\Big(y_{j,k}^i \left[1 - \|\mathbf{x}_j^i - \mathbf{x}_k^i\|_{\boldsymbol{\Sigma}_i}^2\right]\Big) \\
& + \frac{\lambda_1}{2} \sum_{i=1}^{m} \|\boldsymbol{\Sigma}_i\|_F^2 + \frac{\lambda_2}{2} \mathrm{tr}(\tilde{\boldsymbol{\Sigma}} \boldsymbol{\Omega}^{-1} \tilde{\boldsymbol{\Sigma}}^T) \\
\text{s.t.} \quad & \boldsymbol{\Sigma}_i \succeq \mathbf{0} \ \forall i \\
& \tilde{\boldsymbol{\Sigma}} = (\mathrm{vec}(\boldsymbol{\Sigma}_1), \ldots, \mathrm{vec}(\boldsymbol{\Sigma}_m)) \\
& \boldsymbol{\Omega} \succeq 0 \\
& \mathrm{tr}(\boldsymbol{\Omega}) = 1, \qquad\qquad (3)
\end{aligned}$$

where $y_{j,k}^i$ is equal to 1 when $y_j^i = y_k^i$ and -1 otherwise, vec$(\cdot)$ denotes the operator which converts a matrix into a vector in a columnwise manner, and λ_1 and λ_2 are the regularization parameters. $\boldsymbol{\Omega}$ is a task covariance matrix which describes the relationships between tasks and so it is a PSD matrix. The first term in the objective function of problem (3) measures the empirical loss for the training sets of the m tasks, the second term penalizes the complexity of each $\boldsymbol{\Sigma}_i$, and the last term measures the task relationships between all pairs of tasks based on each $\boldsymbol{\Sigma}_i$. The last constraint in (3) is to restrict the scale of $\boldsymbol{\Omega}$ to prevent it from reaching a degenerate solution.

From a probabilistic viewpoint, RDML can be seen as obtaining the maximum a posteriori (MAP) solution of a probabilistic model where the likelihood corresponds to the first term in the objective function of problem (1) and the prior on the metric is Gaussian prior corresponding to the second term. Similar to RDML, our multi-task metric learning is also a MAP solution of a probabilistic model where the likelihood is the same as that in RDML for each task and the prior on the metrics of all tasks is matrix-variate normal distribution [11].

We will prove below that problem (3) is a convex optimization problem by proving that each term in the objective function is convex and each constraint is also convex.

Theorem 1. *Problem (3) is convex with respect to* $\mathbf{W}$, $\mathbf{b}$ *and* $\boldsymbol{\Omega}$.

Proof
It is easy to see that the first two terms in the objective function are convex with respect to (w.r.t.) all variables and the constraints in (3) are also convex. We rewrite the third term as $\mathrm{tr}(\mathbf{W}\boldsymbol{\Omega}^{-1}\mathbf{W}^T) = \sum_t \mathbf{W}(t,:)\boldsymbol{\Omega}^{-1}\mathbf{W}(t,:)^T$ where $\mathbf{W}(t,:)$ is the tth row of $\mathbf{W}$. Since $\mathbf{W}(t,:)\boldsymbol{\Omega}^{-1}\mathbf{W}(t,:)^T$ is a matrix fractional function as in Example 3.4 on page 76 of [3], it is convex w.r.t. $\mathbf{W}(t,:)$ and $\boldsymbol{\Omega}$ when $\boldsymbol{\Omega}$ is a PSD matrix (which is satisfied by the first constraint of (3)). Since $\mathbf{W}(t,:)$ is a row of $\mathbf{W}$, $\mathbf{W}(t,:)\boldsymbol{\Omega}^{-1}\mathbf{W}(t,:)^T$ is also convex w.r.t. $\mathbf{W}$ and $\boldsymbol{\Omega}$. Because the summation operation can preserve convexity according to the analysis on page 79 of [3], $\mathrm{tr}(\mathbf{W}\boldsymbol{\Omega}^{-1}\mathbf{W}^T) = \sum_t \mathbf{W}(t,:)\boldsymbol{\Omega}^{-1}\mathbf{W}(t,:)^T$ is convex w.r.t. $\mathbf{W}$, $\mathbf{b}$ and $\boldsymbol{\Omega}$. So the objective function and the constraints in problem (3) are convex w.r.t. all variables and hence problem (3) is jointly convex. □

Even though problem (3) is convex with respect to $\{\boldsymbol{\Sigma}_i\}$ and $\boldsymbol{\Omega}$ jointly, it is not easy to optimize it with respect to all the variables simultaneously. Here we propose an alternating method to solve the problem more efficiently. Specifically, we first optimize the objective function with respect to $\boldsymbol{\Sigma}_i$ when $\boldsymbol{\Omega}$ and $\{\boldsymbol{\Sigma}\}_{-i} \stackrel{\text{def}}{=} \{\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_{i-1}, \boldsymbol{\Sigma}_{i+1}, \ldots, \boldsymbol{\Sigma}_m\}$ are fixed, and then optimize it with respect to $\boldsymbol{\Omega}$ when $\{\boldsymbol{\Sigma}_i\}$ are fixed. This procedure is repeated until convergence. Since the original optimization problem is convex, the solution found by this alternating procedure is guaranteed to be the globally optimal solution [2].

Because multi-task metric learning is not the focus of this paper, we leave the detailed optimization procedure to Appendix A.

4. TRANSFER METRIC LEARNING

Based on the multi-task metric learning problem formulated in the previous section, we propose a transfer metric learning formulation as a special case which can learn the task relationships between all source tasks and the target task.

Suppose we are given $m-1$ source tasks $\{T_i\}_{i=1}^{m-1}$ and one target task T_m, for $m > 1$. In the target task, the training set contains n_m labeled data points $\{(\mathbf{x}_j^m, y_j^m)\}_{j=1}^{n_m}$. In transfer learning, it is assumed that each source task has enough labeled data and can learn an accurate model with no need to seek help from the other source tasks. So the source tasks are considered to be independent since each source task does not need help from other source tasks. So, similar to the setting in [28], we assume that the metric matrix $\mathbf{\Sigma}_i$ for the ith source task has been learned independently. We hope to use the metric matrices learned to help the learning of the target task because the labeled data there is scarce.

4.1 Optimization Problem

Based on problem (3), we formulate the optimization problem for TML as follows:

$$\begin{aligned}\min_{\mathbf{\Sigma}_m,\mathbf{\Omega}} \quad & \frac{2}{n_m(n_m-1)}\sum_{j<k} g\Big(y^m_{j,k}\left[1-\|\mathbf{x}^m_j-\mathbf{x}^m_k\|^2_{\mathbf{\Sigma}_m}\right]\Big) \\ & +\frac{\lambda_1}{2}\|\mathbf{\Sigma}_m\|_F^2+\frac{\lambda_2}{2}\mathrm{tr}(\tilde{\mathbf{\Sigma}}\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T) \\ \text{s.t.} \quad & \mathbf{\Sigma}_m \succeq \mathbf{0} \\ & \tilde{\mathbf{\Sigma}} = (\mathrm{vec}(\mathbf{\Sigma}_1),\ldots,\mathrm{vec}(\mathbf{\Sigma}_{m-1}),\mathrm{vec}(\mathbf{\Sigma}_m)) \\ & \mathbf{\Omega}\succeq 0 \\ & \mathrm{tr}(\mathbf{\Omega})=1. \end{aligned} \tag{4}$$

Since we assume that the source tasks are independent and each source is of equal importance, we can express $\mathbf{\Omega}$ as

$$\mathbf{\Omega}=\begin{pmatrix}\alpha\mathbf{I}_{m-1} & \boldsymbol{\omega}_m\\ \boldsymbol{\omega}_m^T & \omega\end{pmatrix},$$

where $\mathbf{I}_a$ denotes the $a\times a$ identity matrix, $\boldsymbol{\omega}_m$ denotes the task covariances between the target task and the source tasks, and ω denotes the variance of the target task. According to the last constraint in problem (4), we can get

$$\alpha=\frac{1-\omega}{m-1}.$$

From Theorem 1, it is easy to show that problem (4) is also jointly convex with respect to all variables. Moreover, from the block matrix inversion formula, we can get

$$\begin{aligned}&\mathbf{\Omega}^{-1}\\ =&\begin{pmatrix}\frac{1-\omega}{m-1}\mathbf{I}_{m-1} & \boldsymbol{\omega}_m\\ \boldsymbol{\omega}_m^T & \omega\end{pmatrix}^{-1}\\ =&\begin{pmatrix}\mathbf{I}_{m-1} & \mathbf{a}\\ \mathbf{0}_{m-1}^T & 1\end{pmatrix}\begin{pmatrix}\frac{(m-1)\mathbf{I}_{m-1}}{1-\omega} & \mathbf{0}_{m-1}\\ \mathbf{0}_{m-1}^T & \frac{1}{c}\end{pmatrix}\begin{pmatrix}\mathbf{I}_{m-1} & \mathbf{0}_{m-1}\\ \mathbf{a}^T & 1\end{pmatrix},\end{aligned}$$

where $\mathbf{0}_d$ denotes the $d\times 1$ zero vector, $\mathbf{a}=-\frac{(m-1)\boldsymbol{\omega}_m}{1-\omega}$ and $c=\omega-\frac{(m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m}{1-\omega}$.

Let $\tilde{\mathbf{\Sigma}}_s=(\mathrm{vec}(\mathbf{\Sigma}_1),\ldots,\mathrm{vec}(\mathbf{\Sigma}_{m-1}))$, which is a constant matrix here, denote the parameter matrix of the source tasks. Then we can get

$$\begin{aligned}&\mathrm{tr}(\tilde{\mathbf{\Sigma}}\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T)\\ =&\mathrm{tr}\Big(\Big(\tilde{\mathbf{\Sigma}}_s,\mathrm{vec}(\mathbf{\Sigma}_m)\Big)\mathbf{\Omega}^{-1}\begin{pmatrix}\tilde{\mathbf{\Sigma}}_s^T\\ \mathrm{vec}(\mathbf{\Sigma}_m)^T\end{pmatrix}\Big)\\ =&\mathrm{tr}\Big(\begin{pmatrix}\tilde{\mathbf{\Sigma}}_s^T\\ \mathrm{vec}(\mathbf{\Sigma}_m)^T-\frac{(m-1)}{1-\omega}\boldsymbol{\omega}_m^T\tilde{\mathbf{\Sigma}}_s^T\end{pmatrix}^T\begin{pmatrix}\frac{(m-1)\mathbf{I}_{m-1}}{1-\omega} & \mathbf{0}_{m-1}\\ \mathbf{0}_{m-1}^T & \frac{1}{c}\end{pmatrix}^{-1}\\ &\begin{pmatrix}\tilde{\mathbf{\Sigma}}_s^T\\ \mathrm{vec}(\mathbf{\Sigma}_m)^T-\frac{(m-1)}{1-\omega}\boldsymbol{\omega}_m^T\tilde{\mathbf{\Sigma}}_s^T\end{pmatrix}\Big)\\ =&\frac{m-1}{1-\omega}\mathrm{tr}(\tilde{\mathbf{\Sigma}}_s^T\tilde{\mathbf{\Sigma}}_s)+\frac{1}{c}\|\mathrm{vec}(\mathbf{\Sigma}_m)-\frac{(m-1)}{1-\omega}\tilde{\mathbf{\Sigma}}_s\boldsymbol{\omega}_m\|_2^2\\ =&\frac{(1-\omega)\|\mathbf{\Sigma}_m\|_F^2-2(m-1)\mathrm{vec}(\mathbf{\Sigma}_m)^T\tilde{\mathbf{\Sigma}}_s\boldsymbol{\omega}_m+(m-1)\omega\mathrm{tr}(\tilde{\mathbf{\Sigma}}_s^T\tilde{\mathbf{\Sigma}}_s)}{\omega(1-\omega)-(m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m}.\end{aligned} \tag{5}$$

Moreover, according to the Schur complement [3], we have

$$\mathbf{\Omega}\succeq 0 \iff \omega\geq\frac{m-1}{1-\omega}\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m \text{ and } \frac{(m-1)\mathbf{I}_{m-1}}{1-\omega}\succeq 0,$$

which is equivalent to

$$\mathbf{\Omega}\succeq 0 \iff \omega(1-\omega)\geq(m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m.$$

Then problem (4) can be simplified to

$$\begin{aligned}\min_{\mathbf{\Sigma}_m,\boldsymbol{\omega}_m,\omega,\mathbf{\Omega}} \quad & \frac{2}{n_m(n_m-1)}\sum_{j<k} g\Big(y^m_{j,k}\left[1-\|\mathbf{x}^m_j-\mathbf{x}^m_k\|^2_{\mathbf{\Sigma}_m}\right]\Big) \\ & +\frac{\lambda_1}{2}\|\mathbf{\Sigma}_m\|_F^2+\frac{\lambda_2}{2}\mathrm{tr}(\tilde{\mathbf{\Sigma}}\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T) \\ \text{s.t.} \quad & \mathbf{\Sigma}_m \succeq \mathbf{0} \\ & \mathbf{\Omega}=\begin{pmatrix}\frac{1-\omega}{m-1}\mathbf{I}_{m-1} & \boldsymbol{\omega}_m\\ \boldsymbol{\omega}_m^T & \omega\end{pmatrix} \\ & \tilde{\mathbf{\Sigma}}=(\tilde{\mathbf{\Sigma}}_s,\mathrm{vec}(\mathbf{\Sigma}_m)) \\ & \omega(1-\omega)\geq(m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m, \end{aligned} \tag{6}$$

where the last term in the objective function can be simplified as in Eq. (5).

Compared with the L-DML method in [28], our method has some advantages. First, the formulation of TML is convex and so there is guarantee to find the globally optimal solution. Second, similar to multi-task metric learning proposed in the previous section, TML can model positive, negative and zero task correlations in a unified formulation but L-DML cannot model negative task correlation. As an extreme case, we can deal with the situation in which all source tasks are outlier tasks, but L-DML cannot handle it due to the constraint $\sum_{k=1}^K \mu_k=1$ in problem (2).

Moreover, compared with problem (4), there is no PSD constraint on $\mathbf{\Omega}$ in problem (6) making it simpler than problem (4). In the next section, we will discuss how to solve problem (6).

4.2 Optimization Procedure

As in multi-task metric learning, problem (6) is a convex problem and we still use an alternating method to solve it. Specifically, we first optimize the objective function with respect to $\mathbf{\Sigma}_m$ when $\boldsymbol{\omega}_m$ and ω are fixed, and then optimize it with respect to $\boldsymbol{\omega}_m$ and ω when $\mathbf{\Sigma}_m$ is fixed. This procedure is repeated until convergence. As before, the solution found

Table 1: Online Learning Algorithm for Problem (7)

```
Input: labeled data $(\mathbf{x}_j^m, y_j^m)$ $(j = 1, \ldots, n_m)$, matrix $\mathbf{M}$, $\lambda'_1$, $\lambda'_2$ and predefined learning rate $\eta$
Initialize $\mathbf{\Sigma}_m^{(0)} = \frac{\lambda'_2}{\lambda'_1}\mathbf{M}$;
for $t = 1, \ldots, T_{max}$ do
    Receive a pair of training data points $\{(\mathbf{x}_j^m, y_j^m), (\mathbf{x}_k^m, y_k^m)\}$;
    Compute $y$: $y = 1$ if $y_j^m = y_k^m$, and $y = -1$ otherwise;
    if the training pair $(\mathbf{x}_j^m, \mathbf{x}_k^m), y$ is classified correctly, i.e., $y(1 - \|\mathbf{x}_j^m - \mathbf{x}_k^m\|^2_{\mathbf{\Sigma}_m^{(t-1)}}) > 0$ then
        $\mathbf{\Sigma}_m^{(t)} = \mathbf{\Sigma}_m^{(t-1)}$;
    else if $y == -1$
        $\mathbf{\Sigma}_m^{(t)} = \mathbf{\Sigma}_m^{(t-1)} + \eta(\mathbf{x}_j^m - \mathbf{x}_k^m)(\mathbf{x}_j^m - \mathbf{x}_k^m)^T$;
    else
        $\mathbf{\Sigma}_m^{(t)} = \pi_{S_+}\Big(\mathbf{\Sigma}_m^{(t-1)} - \eta(\mathbf{x}_j^m - \mathbf{x}_k^m)(\mathbf{x}_j^m - \mathbf{x}_k^m)^T\Big)$ where $\pi_{S_+}(\mathbf{A})$ projects matrix $\mathbf{A}$ into the positive
        semidefinite cone;
    end if
end for
Output: metric $\mathbf{\Sigma}_m^{(T_{max})}$
```

by this alternating procedure is globally optimal [2]. In what follows, we will present the two subproblems separately.

Optimizing w.r.t. $\mathbf{\Sigma}_m$ when $\boldsymbol{\omega}_m$ and ω are fixed

Utilizing Eq. (5), the optimization problem with respect to $\mathbf{\Sigma}_m$ is formulated as

$$\begin{aligned}\min_{\mathbf{\Sigma}_m} \quad & \frac{2}{n_m(n_m-1)}\sum_{j<k} g\Big(y_{j,k}^m\left[1 - \|\mathbf{x}_j^m - \mathbf{x}_k^m\|^2_{\mathbf{\Sigma}_m}\right]\Big) \\ & + \frac{\lambda'_1}{2}\|\mathbf{\Sigma}_m\|_F^2 - \lambda'_2 \mathrm{tr}(\mathbf{\Sigma}_m^T\mathbf{M}) \\ \text{s.t.} \quad & \mathbf{\Sigma}_m \succeq \mathbf{0}, \end{aligned} \tag{7}$$

where

$$\begin{aligned}\lambda'_1 &= \lambda_1 + \frac{\lambda_2(1-\omega)}{\omega(1-\omega) - (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m}, \\ \lambda'_2 &= \frac{\lambda_2(m-1)}{\omega(1-\omega) - (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m},\end{aligned}$$

$\mathbf{M}$ is a matrix such that $\mathrm{vec}(\mathbf{M}) = \tilde{\mathbf{\Sigma}}_s\boldsymbol{\omega}_m$. It is easy to show that $\mathbf{M}$ is a combination of $\mathbf{\Sigma}_i$ $(i = 1, \ldots, m-1)$ as $\mathbf{M} = \sum_{i=1}^{m-1}\omega_{mi}\mathbf{\Sigma}_i$ where ω_{mi} is the ith element of $\boldsymbol{\omega}_m$.

Similar to [13], we can use an online learning method to solve problem (7) and the algorithm is depicted in Table 1. This algorithm is similar to that in [13] except the initial step for $\mathbf{\Sigma}_m^{(0)}$. In [13], the initial value for $\mathbf{\Sigma}_m^{(0)}$ is a zero matrix but here it is $\frac{\lambda'_2}{\lambda'_1}\mathbf{M}$. Note that $\mathbf{M}$ is a combination of the metrics learned from the source tasks where each combination weight is the task covariance between a source task and the target task. This agrees with our intuition that a positively correlated source task will have a large weight on the initial value for $\mathbf{\Sigma}_m$, an outlier task has negligle contribution and a negatively correlated task even has opposite effect.

Optimizing w.r.t. $\boldsymbol{\omega}_m$ and ω when $\mathbf{\Sigma}_m$ is fixed

Utilizing Eq. (5), the optimization problem with respect to $\boldsymbol{\omega}_m$ and ω is formulated as

$$\begin{aligned}\min_{\boldsymbol{\omega}_m, \omega, \mathbf{\Omega}} \quad & \mathrm{tr}(\tilde{\mathbf{\Sigma}}\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T) \\ \text{s.t.} \quad & \mathbf{\Omega} = \begin{pmatrix} \frac{1-\omega}{m-1}\mathbf{I}_{m-1} & \boldsymbol{\omega}_m \\ \boldsymbol{\omega}_m^T & \omega \end{pmatrix} \\ & \omega(1-\omega) \geq (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m. \end{aligned} \tag{8}$$

We impose a constraint as $\tilde{\mathbf{\Sigma}}\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T \preceq \frac{1}{t}\mathbf{I}_{d^2}$ and the objective function becomes $\min \frac{1}{t}$. Using the Schur complement, we can get

$$\tilde{\mathbf{\Sigma}}\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T \preceq \frac{1}{t}\mathbf{I}_{d^2} \Longleftrightarrow \begin{pmatrix} \mathbf{\Omega} & \tilde{\mathbf{\Sigma}}^T \\ \tilde{\mathbf{\Sigma}} & \frac{1}{t}\mathbf{I}_{d^2} \end{pmatrix} \succeq \mathbf{0}.$$

By using the Schur complement again, we get

$$\begin{pmatrix} \mathbf{\Omega} & \tilde{\mathbf{\Sigma}}^T \\ \tilde{\mathbf{\Sigma}} & \frac{1}{t}\mathbf{I}_{d^2} \end{pmatrix} \succeq \mathbf{0} \Longleftrightarrow \mathbf{\Omega} - t\tilde{\mathbf{\Sigma}}^T\tilde{\mathbf{\Sigma}} \succeq \mathbf{0}.$$

We write $\tilde{\mathbf{\Sigma}}^T\tilde{\mathbf{\Sigma}} = \begin{pmatrix} \mathbf{\Psi}_{11} & \mathbf{\Psi}_{12} \\ \mathbf{\Psi}_{12}^T & \Psi_{22} \end{pmatrix}$ where $\mathbf{\Psi}_{11} \in \mathbb{R}^{(m-1)\times(m-1)}$, $\mathbf{\Psi}_{12} \in \mathbb{R}^{(m-1)\times 1}$ and $\Psi_{22} \in \mathbb{R}$. Then $\mathbf{\Omega} - t\tilde{\mathbf{\Sigma}}^T\tilde{\mathbf{\Sigma}} \succeq \mathbf{0}$ is equivalent to

$$\begin{aligned}& \frac{1-\omega}{m-1}\mathbf{I}_{m-1} - t\mathbf{\Psi}_{11} \succeq \mathbf{0} \\ & \omega - t\Psi_{22} \geq (\boldsymbol{\omega}_m - t\mathbf{\Psi}_{12})^T\left(\frac{1-\omega}{m-1}\mathbf{I}_{m-1} - t\mathbf{\Psi}_{11}\right)^{-1} \\ & \qquad (\boldsymbol{\omega}_m - t\mathbf{\Psi}_{12}).\end{aligned}$$

Let $\mathbf{U}$ and $\lambda_1, \ldots, \lambda_{m-1}$ denote the eigenvector matrix and eigenvalues of $\mathbf{\Psi}_{11}$ with $\lambda_1 \geq \ldots \geq \lambda_{m-1} \geq 0$. Then

$$\frac{1-\omega}{m-1}\mathbf{I}_{m-1} - t\mathbf{\Psi}_{11} \succeq \mathbf{0} \Longleftrightarrow \frac{1-\omega}{m-1} \geq \lambda_1 t$$

and

$$\begin{aligned}& \left(\frac{1-\omega}{m-1}\mathbf{I}_{m-1} - t\mathbf{\Psi}_{11}\right)^{-1} \\ = \quad & \mathbf{U}\,\mathrm{diag}\left(\frac{1-\omega}{m-1} - t\lambda_1, \ldots, \frac{1-\omega}{m-1} - t\lambda_{m-1}\right)\mathbf{U}^T.\end{aligned}$$

Combining the above results, problem (8) is formulated as

$$\begin{aligned}\min_{\boldsymbol{\omega}_m,\omega,\mathbf{f},t} \quad & -t \\ \text{s.t.} \quad & \frac{1-\omega}{m-1} \geq t\lambda_1 \\ & \mathbf{f} = \mathbf{U}^T(\boldsymbol{\omega}_m - t\boldsymbol{\Psi}_{12}) \\ & \sum_{j=1}^{m-1} \frac{f_j^2}{\frac{1-\omega}{m-1} - t\lambda_j} \leq \omega - t\Psi_{22} \\ & \omega(1-\omega) \geq (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m, \qquad (9)\end{aligned}$$

where f_j is the jth element of $\mathbf{f}$. By introducing new variables h_j and r_j $(j = 1, \ldots, m-1)$, (9) is reformulated as

$$\begin{aligned}\min_{\boldsymbol{\omega}_m,\omega,\mathbf{f},t,\{h_j\},\{r_j\}} \quad & -t \\ \text{s.t.} \quad & \frac{1-\omega}{m-1} \geq t\lambda_1 \\ & \mathbf{f} = \mathbf{U}^T(\boldsymbol{\omega}_m - t\boldsymbol{\Psi}_{12}) \\ & \sum_{j=1}^{m-1} h_j \leq \omega - t\Psi_{22} \\ & r_j = \frac{1-\omega}{m-1} - t\lambda_j \ \forall j \\ & \frac{f_j^2}{r_j} \leq h_j \ \forall j \\ & \omega(1-\omega) \geq (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m. \quad (10)\end{aligned}$$

Since

$$\frac{f_j^2}{r_j} \leq h_j \iff \left\| \begin{pmatrix} f_j \\ \frac{r_j - h_j}{2} \end{pmatrix} \right\|_2 \leq \frac{r_j + h_j}{2}$$

and

$$\omega(1-\omega) \geq (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m \iff \left\| \begin{pmatrix} \sqrt{m-1}\boldsymbol{\omega}_m \\ \frac{\omega-1}{2} \\ \omega \end{pmatrix} \right\|_2 \leq \frac{\omega+1}{2},$$

problem (10) is a second-order cone programming (SOCP) problem [15] with $O(m)$ variables and $O(m)$ constraints. In many applications, m is very small and we can use a standard solver to solve problem (10) very efficiently.

We set the initial value of ω to $\frac{1}{m}$ and that of $\boldsymbol{\omega}_m$ to a zero vector which corresponds to the assumption that the target task is unrelated to the source tasks.

After learning the optimal values of $\boldsymbol{\Sigma}_m$, we can make prediction for a new data point. Given a test data point $\mathbf{x}_\star^m$ for the target task T_m, we first calculate the distances between $\mathbf{x}_\star^m$ and all training data points in T_m based on the learned metric $\boldsymbol{\Sigma}_m$ and then use the k-nearest neighbor classifier to classify $\mathbf{x}_\star^m$, where we choose $k = 1$ for simplicity in our experiments.

5. EXPERIMENTS

We study TML empirically in this section by comparing it with two metric learning methods, ITML[2] [8] and RDML [13], and another metric learning method for transfer learning, L-DML [28]. We use the CVX solver [10][3] to solve problem (10). We set the learning rate η in Table 1 to 0.01. For ITML, RDML and L-DML, the best parameters reported in [8, 13, 28] are used.

5.1 Wine Quality Classification

The wine dataset[4] is about wine quality including red and white wine samples. The features include objective tests (e.g., PH values) and the output is based on sensory data. The labels are given by experts with grades between 0 (very bad) and 10 (very excellent). There are 1599 records for the red wine and 4898 for the white wine and so there are two tasks, one for red wine classification and the other for white wine classification. Each task is treated as the target task and the other task as the source task. To see the effect of varying the size of the training set, we vary the percentage of the training data used from 5% to 20%. Each configuration is repeated 10 times. The mean and standard deviation of the classification accuracy are reported in Fig. 1(a) and 1(b). From the results, we can see that the performance of L-DML is comparable with that of ITML and RDML and TML is always the best one for both tasks.

5.2 Handwritten Letter Classification

The handwritten letter classification applicaton[5] consists of seven tasks where each task is a binary classification problem. The corresponding letters for each task are: c/e, g/y, m/n, a/g, a/o, f/t and h/n. Each data point has 128 features corresponding to the pixel values of the handwritten letter images. For each task, there are about 1000 positive and 1000 negative data points. The experimental settings are the same as those for wine quality classification above. The results are plotted in Fig. 2(a) to 2(g). From the results, we find that the performance of L-MDL is worse than that of ITML and RDML on some tasks (4th, 6th and 7th tasks). This may be due to the fact that the objective function of L-MDL is non-convex and hence it is easy to get trapped in bad local minima. TML shows the best performance on almost every task.

5.3 USPS Digit Classification

The USPS digit dataset[5] contains 7291 examples each of 255 features. There are nine classification tasks, each corresponding to the classification of two digits. The experimental settings are the same as those for handwritten letter classification. The results are reported in Fig. 3(a) to 3(i). Similar to handwritten digit classification, L-MDL is worse than ITML and RDML on some tasks and TML is better than other methods on almost all tasks.

6. CONCLUSION

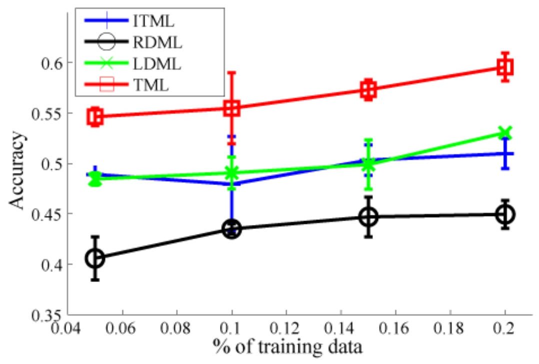
In this paper, we have proposed a transfer metric learning method to alleviate the labeled data deficiency problem in the target learning task by exploiting useful information from some source tasks. The learning of the distance metrics from the source tasks and the relationships between the source tasks and the target task is formulated as a convex optimization problem which can be solved efficiently. In our future research, we will extend TML to the semi-supervised setting by exploiting useful information contained in the unlabeled data as well.

[2]The implementation of ITML can be found in http://www.cs.utexas.edu/users/pjain/itml/.

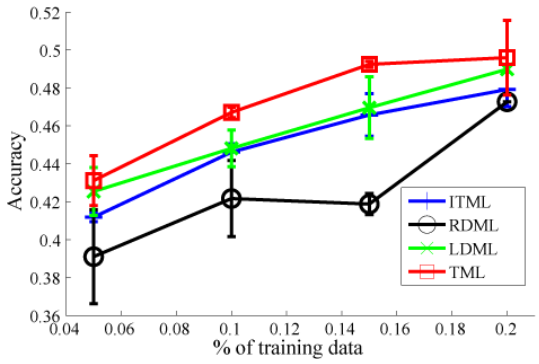
[3]http://stanford.edu/∼boyd/cvx

[4]http://archive.ics.uci.edu/ml/datasets/Wine+Quality

[5]http://multitask.cs.berkeley.edu/

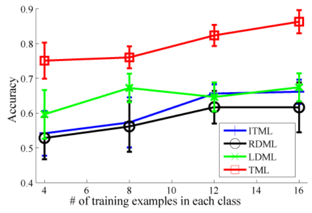


(a) 1st task

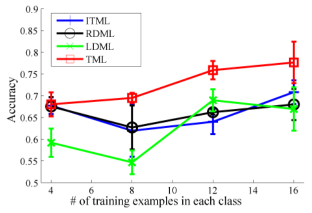


(b) 2nd task

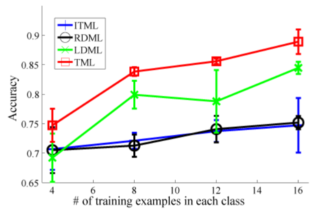
Figure 1: Overall performance on wine quality classification application.



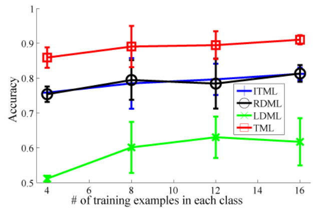
(a) 1st task



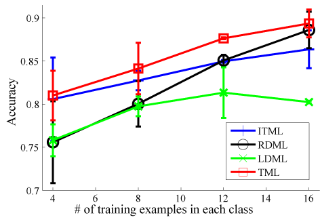
(b) 2nd task



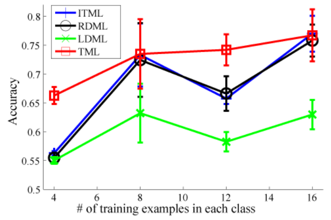
(c) 3rd task



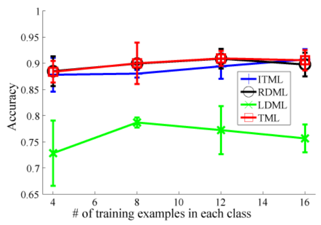
(d) 4th task



(e) 5th task



(f) 6th task



(g) 7th task

Figure 2: Overall performance on handwritten letter classification application when one task is the target and the others are source tasks.

Acknowledgments

This research has been supported by General Research Fund 622209 from the Research Grants Council of Hong Kong.

7. REFERENCES

[1] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[2] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, 2004.

[4] R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[5] 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*, Banff, Alberta, Canada, 2004.

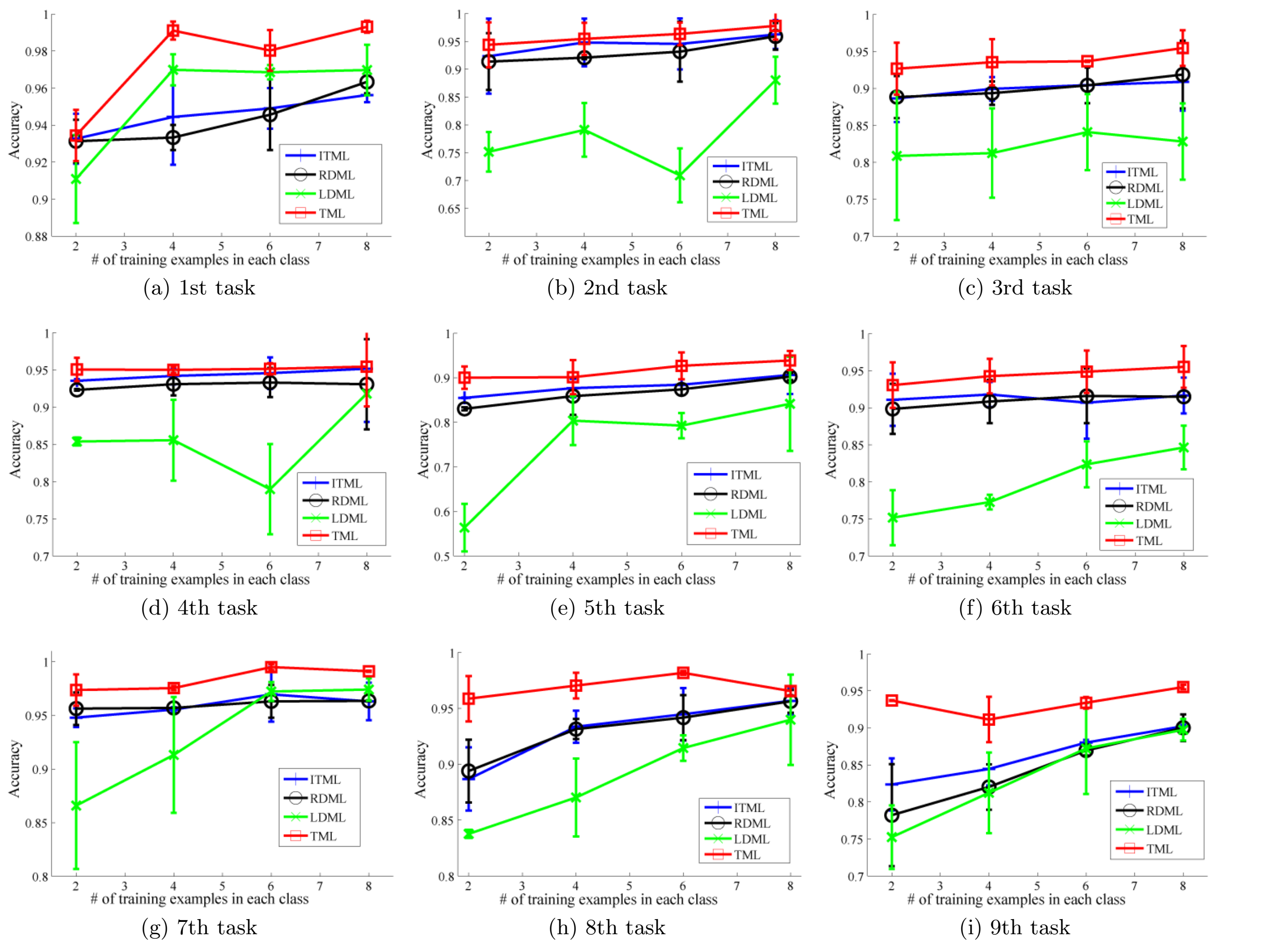
[6] J. Chen, Z. Zhao, J. Ye, and H. Liu. Nonlinear adaptive distance metric learning for clustering. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 123–132, San Jose, California, USA, 2007.

[7] J. V. Davis and I. S. Dhillon. Structured metric learning for high dimensional problems. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 195–203, Las Vegas, Nevada, USA, 2008.

[8] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *Proceedings of the Twenty-Fourth International Conference on Machine Learning*, pages 209–216, Corvalis, Oregon, USA, 2007.

[9] T. Evgeniou and M. Pontil. Regularized multi-task learning. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 109–117, Seattle, Washington, USA, 2004.

[10] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming (web page and software), June 2009.



(a) 1st task (b) 2nd task (c) 3rd task
(d) 4th task (e) 5th task (f) 6th task
(g) 7th task (h) 8th task (i) 9th task

Figure 3: Overall performance on USPS digit classification application.

[11] A. K. Gupta and D. K. Nagar. *Matrix Variate Distributions*. Chapman & Hall, 2000.

[12] T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(6):607–616, 1996.

[13] R. Jin, S. Wang, and Y. Zhou. Regularized distance metric learning: Theory and algorithm. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 862–870, Vancouver, British Columbia, Canada, 2009.

[14] W. Kienzle and K. Chellapilla. Personalized handwriting recognition via biased regularization. In *Proceedings of the Twenty-Third International Conference on Machine Learning*, pages 457–464, 2006.

[15] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228, 1998.

[16] S. J. Pan, J. T. Kwok, and Q. Yang. Transfer learning via dimensionality reduction. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, pages 677–682, Chicago, Illinois, USA, 2008.

[17] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1187–1192, Pasadena, California, USA, 2009.

[18] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 2009.

[19] M. T. Rosenstein, Z. Marx, and L. P. Kaelbling. To transfer or not to transfer. In *NIPS-05 Workshop on Inductive Transfer: 10 Years Later*, 2005.

[20] S. Thrun. Is learning the *n*-th thing any easier than learning the first? In D. S. Touretzky, M. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 640–646, Denver, CO, 1996.

[21] Z. Wang, Y. Song, and C. Zhang. Transferred dimensionality reduction. In *Proceedings of European*

Conference on Machine Learning and Knowledge Discovery in Databases, pages 550–565, Antwerp, Belgium, 2008.

[22] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1473–1480, Vancouver, British Columbia, Canada, 2005.

[23] K. Q. Weinberger and L. K. Saul. Fast solvers and efficient implementations for distance metric learning. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning*, pages 1160–1167, Helsinki, Finland, 2008.

[24] B. Xiao, X. Yang, Y. Xu, and H. Zha. Learning distance metric for regression by semidefinite programming with application to human age estimation. In *Proceedings of the 17th ACM International Conference on Multimedia*, pages 451–460, 2009.

[25] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. J. 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, Vancouver, British Columbia, Canada, 2002.

[26] D.-Y. Yeung and H. Chang. A kernel approach for semisupervised metric learning. *IEEE Transactions on Neural Networks*, 18(1):141–149, 2007.

[27] D.-Y. Yeung, H. Chang, and G. Dai. A scalable kernel-based semi-supervised metric learning algorithm with out-of-sample generalization ability. *Neural Computation*, 20(11):2839–2861, 2008.

[28] Z.-J. Zha, T. Mei, M. Wang, Z. Wang, and X.-S. Hua. Robust distance metric learning with auxiliary knowledge. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1327–1332, Pasadena, California, USA, 2009.

[29] D.-C. Zhan, M. Li, Y.-F. Li, and Z.-H. Zhou. Learning instance specific distances using metric propagation. In *Proceedings of the 26th International Conference on Machine Learning*, pages 1225–1232, Montreal, Quebec, Canada, 2009.

APPENDIX

A. OPTIMIZATION PROCEDURE FOR PROBLEM (3)

We present here the optimization procedure for solving problem (3). We use an alternating method with two subproblems to be presented separately below.

Optimizing w.r.t. $\mathbf{\Sigma}_i$ when $\mathbf{\Omega}$ and $\{\mathbf{\Sigma}\}_{-i}$ are fixed

We first define $\tilde{\mathbf{\Sigma}}$ and $\mathbf{\Omega}^{-1}$ as

$$\begin{aligned}\tilde{\mathbf{\Sigma}} &= \Big(\mathrm{vec}(\mathbf{\Sigma}_i), \tilde{\mathbf{\Sigma}}_{-i}\Big)\\ \mathbf{\Omega}^{-1} &= \begin{pmatrix} \gamma_{ii} & \boldsymbol{\gamma}_i^T \\ \boldsymbol{\gamma}_i & \mathbf{\Gamma}_{-i}\end{pmatrix}.\end{aligned}$$

Then the third term in the objective function of problem (3) can be rewritten as

$$\begin{aligned}&\frac{\lambda_2}{2}\mathrm{tr}(\tilde{\mathbf{\Sigma}}\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T)\\ =&\frac{\lambda_2}{2}\mathrm{tr}\Big((\mathrm{vec}(\mathbf{\Sigma}_i), \tilde{\mathbf{\Sigma}}_{-i})\begin{pmatrix} \gamma_{ii} & \boldsymbol{\gamma}_i^T \\ \boldsymbol{\gamma}_i & \mathbf{\Gamma}_{-i}\end{pmatrix}\begin{pmatrix}\mathrm{vec}(\mathbf{\Sigma}_i)^T \\ \tilde{\mathbf{\Sigma}}_{-i}^T\end{pmatrix}\Big)\\ =&\frac{\lambda_2}{2}\Big(\gamma_{ii}\|\mathrm{vec}(\mathbf{\Sigma}_i)\|_2^2 + 2\boldsymbol{\gamma}_i^T\tilde{\mathbf{\Sigma}}_{-i}^T\mathrm{vec}(\mathbf{\Sigma}_i) + \mathrm{tr}(\tilde{\mathbf{\Sigma}}_{-i}\mathbf{\Gamma}_{-i}\tilde{\mathbf{\Sigma}}_{-i}^T)\Big)\\ =&\frac{\lambda_2}{2}\Big(\gamma_{ii}\|\mathbf{\Sigma}_i\|_F^2 + 2\mathrm{tr}(\mathbf{M}\mathbf{\Sigma}_i) + \mathrm{tr}(\tilde{\mathbf{\Sigma}}_{-i}\mathbf{\Gamma}_{-i}\tilde{\mathbf{\Sigma}}_{-i}^T)\Big),\end{aligned}$$

where $\|\cdot\|_2$ denotes the 2-norm of a vector and $\mathbf{M}$ is a matrix such that $\mathrm{vec}(\mathbf{M}) = \tilde{\mathbf{\Sigma}}_{-i}\boldsymbol{\gamma}_i$. Note that the third term in the last equation above is independent of $\mathbf{\Sigma}_i$. It is easy to show that $\mathbf{M}$ is a symmetric matrix. The optimization problem with respect to $\mathbf{\Sigma}_i$ becomes

$$\begin{aligned}\min_{\mathbf{\Sigma}_i}\quad & \frac{2}{n_i(n_i-1)}\sum_{j<k} g\Big(y_{j,k}^i\left[1 - \|\mathbf{x}_j^i - \mathbf{x}_k^i\|_{\mathbf{\Sigma}_i}^2\right]\Big)\\ & + \frac{\lambda_1 + \lambda_2\gamma_{ii}}{2}\|\mathbf{\Sigma}_i\|_F^2 + \lambda_2\mathrm{tr}(\mathbf{M}\mathbf{\Sigma}_i)\\ \mathrm{s.t.}\quad & \mathbf{\Sigma}_i \succeq \mathbf{0}. \end{aligned}\tag{11}$$

It is easy to see that this problem is a convex semidefinite programming (SDP) problem since the objective function is convex with respect to $\mathbf{\Sigma}_i$ and the constraint is a PSD constraint on $\mathbf{\Sigma}_i$. Even though solving an SDP problem is computationally demanding with poor scalability, we can adopt the technique in [23] and use gradient projection to solve it.

Optimizing w.r.t. $\mathbf{\Omega}$ when $\{\mathbf{\Sigma}_i\}$ are fixed

When $\{\mathbf{\Sigma}_i\}$ are fixed, the optimization problem for finding $\mathbf{\Omega}$ becomes

$$\begin{aligned}\min_{\mathbf{\Omega}}\quad & \mathrm{tr}(\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T\tilde{\mathbf{\Sigma}})\\ \mathrm{s.t.}\quad & \mathbf{\Omega}\succeq 0\\ & \mathrm{tr}(\mathbf{\Omega}) = 1.\end{aligned}\tag{12}$$

Then we have

$$\begin{aligned}\mathrm{tr}(\mathbf{\Omega}^{-1}\mathbf{A}) &= \mathrm{tr}(\mathbf{\Omega}^{-1}\mathbf{A})\mathrm{tr}(\mathbf{\Omega})\\ &= \mathrm{tr}((\mathbf{\Omega}^{-\frac{1}{2}}\mathbf{A}^{\frac{1}{2}})(\mathbf{A}^{\frac{1}{2}}\mathbf{\Omega}^{-\frac{1}{2}}))\mathrm{tr}(\mathbf{\Omega}^{\frac{1}{2}}\mathbf{\Omega}^{\frac{1}{2}})\\ &\geq (\mathrm{tr}(\mathbf{\Omega}^{-\frac{1}{2}}\mathbf{A}^{\frac{1}{2}}\mathbf{\Omega}^{\frac{1}{2}}))^2 = (\mathrm{tr}(\mathbf{A}^{\frac{1}{2}}))^2,\end{aligned}$$

where $\mathbf{A} = \tilde{\mathbf{\Sigma}}^T\tilde{\mathbf{\Sigma}}$. The first equality holds because of the last constraint in problem (12) and the last inequality holds because of the Cauchy-Schwarz inequality for the Frobenius norm. Moreover, $\mathrm{tr}(\mathbf{\Omega}^{-1}\mathbf{A})$ attains its minimum value $(\mathrm{tr}(\mathbf{A}^{\frac{1}{2}}))^2$ if and only if

$$\mathbf{\Omega}^{-\frac{1}{2}}\mathbf{A}^{\frac{1}{2}} = a\mathbf{\Omega}^{\frac{1}{2}}$$

for some constant a and $\mathrm{tr}(\mathbf{\Omega}) = 1$. So we can get the following analytical solution:

$$\mathbf{\Omega} = \frac{\left(\tilde{\mathbf{\Sigma}}^T\tilde{\mathbf{\Sigma}}\right)^{\frac{1}{2}}}{\mathrm{tr}\left(\left(\tilde{\mathbf{\Sigma}}^T\tilde{\mathbf{\Sigma}}\right)^{\frac{1}{2}}\right)}.$$