Multi-task Multi-view Learning for Heterogeneous Tasks

Xin Jin[1,2], Fuzhen Zhuang[1], Hui Xiong[3], Changying Du[1,2], Ping Luo[1] and Qing He[1]

[1]Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, {jinx, zhuangfz, ducy, heq}@ics.ict.ac.cn, luop@ict.ac.cn
[2]University of Chinese Academy of Sciences, Beijing 100049, China
[3]MSIS Department, Rutgers University, hxiong@rutgers.edu

ABSTRACT

Multi-task multi-view learning deals with the learning scenarios where multiple tasks are associated with each other through multiple shared feature views. All previous works for this problem assume that the tasks use the same set of class labels. However, in real world there exist quite a few applications where the tasks with several views correspond to different set of class labels. This new learning scenario is called *Multi-task Multi-view Learning for Heterogeneous Tasks* in this study. Then, we propose a Multi-tAsk MUlti-view Discriminant Analysis (MAMUDA) method to solve this problem. Specifically, this method collaboratively learns the feature transformations for different views in different tasks by exploring the shared task-specific and problem intrinsic structures. Additionally, MAMUDA method is convenient to solve the multi-class classification problems. Finally, the experiments on two real-world problems demonstrate the effectiveness of MAMUDA for heterogeneous tasks.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning—*Machine Learning*

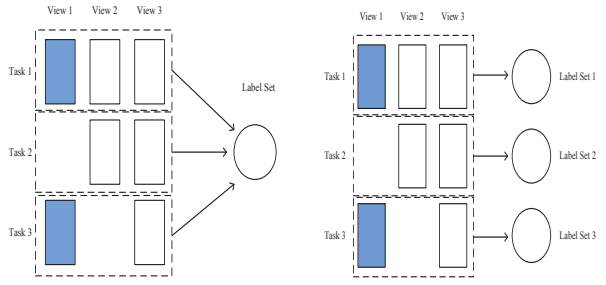
Keywords

Multi-task Learning; Multi-view Learning; Heterogeneous Tasks; Discriminant Analysis; Multi-class Classification

1. INTRODUCTION

Many real-world problems exhibit dual-diversity. Indeed, it is often to see that a single learning task has features in multiple views. This is known as multi-view learning. Also, there are multi-task learning scenarios, where different learning tasks might be related with each other through one or more shared views (features). For example, the task of classifying web pages from Yahoo [1] is related with the task to classify web pages from Open Directory Project [2]. Meanwhile, we can obtain three views of features for a given web page including the content of the web page, the anchor text attached to hyperlinks pointing to this we bpage, and the link structure of all linked web pages. Another example is the music classification problem. As we know, classifying English songs and Chinese songs are two related tasks, and both tasks have audio features. However, these two tasks also have task-specific features, such as Chinese song lyric and English song lyric. Such type of problems are widely known as Multi-task Multi-view (MTMV) learning problems [14, 26, 16].

The traditional multi-task learning (MTL) [5, 6, 7, 22] or multi-view learning (MVL) [4, 8, 21] methods are not designed for the MTMV problem. There are MTMV learning methods [14, 26, 16], which can make a good use of the information contained in multiple tasks and multiple views. However, they all have the assumption that the multiple classification tasks have the same set of class labels, and they are designed for binary classification problems. In many real-world applications, multiple tasks often do not share the same set of labels. For example, in the web page classification problem, the categories in Yahoo and the Open Directory Project are not the same. In other words, each task has a task-specific class label set. Similarly, for the music classification problem, English songs usually have different categories from Chinese songs because of the difference between western and eastern culture. In this paper, we call this type of problem as the MTMV problem with heterogeneous tasks. In contrast, the traditional MTMV problem, in which all tasks share the same label set, is treated as the MTMV problem with homogeneous tasks. Figure 1 shows the difference of these two types of problems.



(a) *Homogeneous Tasks* (b) *Heterogeneous Tasks*

Figure 1: Two Types of MTMV Problems

[1]http://www.yahoo.com/

[2]http://www.dmoz.org/

CIKM'14, November 03–07, 2014, Shanghai, China.

http://dx.doi.org/10.1145/2661829.2662054 .

Existing MTMV learning methods [14, 26, 16] are not designed for MTMV problems with heterogeneous tasks, since they assume all the tasks have the same class label set and they share knowledge among multiple tasks by sharing some class-dependent model parameters. Also, they are binary classification methods, which require nontrivial extensions in order for them to handle multi-class problems, especially when the number of classes is large.

To this end, in this paper, we propose a Multi-tAsk MUltiview Discriminant Analysis (MAMUDA) learning method. Since multiple views of features exist in the problem, it is difficult to directly share knowledge through the original feature space. To facilitate information sharing, this paper extends the classical LDA method [12] and collectively learns the feature transformation matrices for all the views from each task. The classical LDA model tries to transform a feature vector x (row vector) in the original feature space into a vector x' in the *discriminant feature space*, through which the data become more separable. This process is explicitly shown in Figure 2, which also can be written in the equation $xW = x'$. Here, W is the matrix for feature transformation.

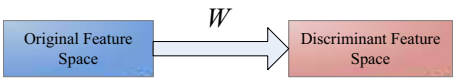


Figure 2: LDA Feature Transformation Process

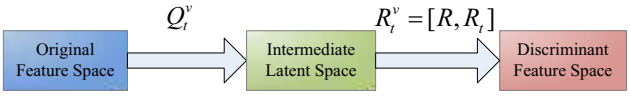


Figure 3: MAMUDA Feature Transformation Process

In our method, the transformation is divided into two steps, which is shown in Figure 3. First, through the transformation matrix Q_t^v, a data sample x from the view v in task t is transformed into an *intermediate latent space*. $\{Q_t^v\}$ are dependent on the views and tasks, thus they are different for different views and tasks. Through $\{Q_t^v\}$, all the views from all the tasks are transformed into a common intermediate latent space that is shared by all the views from each task. Then, through $\{R_t^v\}$, an instance from each view of each task is transformed from the common intermediate latent space into their corresponding discriminant space. R_t^v contains two parts of information. One is R, which is shared by all the views from every task. The other part R_t is shared by the views for a specific task t. They help the knowledge sharing among tasks and views.

With these assumptions we formulate an optimization problem which collaboratively learns the feature transformations of the data from each view and each task. An alternating optimization algorithm is proposed to solved the problem, where each subproblem can be guaranteed to achieve global optimality. With the transformation matrices Q_t^v and R_t^v, the data can be transformed into the discriminant space. Namely, for a data vector x from the view v and task t, its corresponding discriminant space representation is $xQ_t^vR_t^v$. Then, the nearest neighbor classifier is used to make prediction in the discriminant feature space.

In summary, our method has several advantages. First, it does not require that multiple tasks share the same class label set and can solve the MTMV problem with heterogeneous tasks. Second, by using a simple nearest neighbor classifier in the final discriminant feature space, it is very convenient to solve multi-class classification problems. Finally, it can deal with the scenarios when some views may be missing in some tasks.

2. PROBLEM BACKGROUND

Here, we first define the MTMV problem with heterogeneous tasks. Then, we introduce some preliminaries.

2.1 MTMV Problem Def nition

Notations. Let $[N : M]$ $(M > N)$ denote a set of integers in the range of N to M inclusively, $\mathrm{tr}(X)$ be the trace of matrix X, X^{-1} be the inverse of X, $\|\cdot\|$ denote the Frobenius norm of a matrix, and I_l be the $l \times l$ identity matrix. Unless specified otherwise, all vectors are column vectors.

The MTMV problem definition is very similar to [26, 16], except that our formulation is more flexible as we do not restrict that all the tasks have the same set of class labels. Assume that the problem includes T tasks and V views in total. For each task $t \in [1 : T]$, there are n_t labeled examples, thus we have $N = \sum_t n_t$. Let d_v be the number of features in the view $v \in [1 : V]$, and the total number of features $D = \sum_v d_v$.

Feature matrix $X_t^v \in \mathbb{R}^{n_t \times d_v}$ is used to denote the labeled samples in task t for view v, each row represents a sample. Let $Y_t \in [1 : C_t]^{n_t \times 1}$ be the label vector of the labeled examples in task t, where C_t is the number of classes in task t. In the ideal situation, every task has features from all the V views. However, in reality, it is common that, in some applications, not all tasks have features available from all the V views, so an indicator matrix $I_{id} \in \{1, 0\}^{T \times V}$ is used to mark which view is missing from which task; that is, if the task t contains view v then $I_{id}(t, v) = 1$, and 0 otherwise. Using this notation, we only consider the "structured" missing views [26] in the sense that, if a view is present in a task, it is present in all the samples in the task; if a view is missing from a task, it is missing in all the samples in the task. Throughout the paper, we use subscripts to denote tasks and superscripts to denote views.

The goal of this paper is to leverage the information from all the tasks and all the views to help each other classify the unlabeled test samples in each task.

2.2 Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) [12] is a popular supervised dimensionality reduction technique in pattern recognition and machine learning. Traditionally, LDA is used for single learning tasks with single view data. Assume that there are N labeled training samples, represented by feature matrix $X \in \mathbb{R}^{N \times d}$ and label vector $Y \in [1 : C]^{N \times 1}$, where each row vector x_i in X denotes a sample and its label y_i is the i-th element of vector Y, d is the number of features and C is the number of classes. There are N_c samples in c-th class, i.e., $\sum_{c=1}^{C} N_c = N$. Let us define between-class scatter matrix $S_b = \sum_{c=1}^{C} \frac{N_c}{N} (\bar{m}_c - \bar{m})^\top (\bar{m}_c - \bar{m})$, within-class scatter matrix $S_w = \sum_{c=1}^{C} \sum_{y_i=c} \frac{1}{N} (x_i - \bar{m}_c)^\top (x_i - \bar{m}_c)$ and total scatter matrix $S_h = \sum_{i=1}^{N} \frac{1}{N} (x_i - \bar{m})^\top (x_i - \bar{m})$, where $\bar{m} = (\sum_{i=1}^{N} x_i)/N$ is the sample mean for the whole training

set and $\bar{m}_c = (\sum_{y_i=c} x_i)/N_c$ is the class mean of the c-th class. It can be easily verified that $S_h = S_b + S_w$. There are two types of objective functions for LDA that are widely used. The first one is the ratio trace form [12]:

$$W^* = \arg\max_W \text{tr}\left((W^\top S_w W)^{-1} W^\top S_b W\right), \quad (1)$$

and the second one is in the trace ratio form [23]:

$$W^* = \arg\max_{W^\top W = I_l} \frac{\text{tr}(W^\top S_b W)}{\text{tr}(W^\top S_h W)}, \quad (2)$$

where l is the reduced dimensionality of the trace ratio form, $W \in \mathbb{R}^{d\times l}$ is the transformation matrix for dimension reduction. The solution of the ratio trace form can be obtained by computing the eigenvectors of the matrix $S_w^{-1}S_b$, while the trace ratio form has no analytical solution and has to resort to an iterative method to obtain the optimal solution. However, the trace ratio form has a much clearer physical meaning. The numerator and denominator of the objective function in the trace ratio form represent the average between-class distance and average total distance in the low-dimensional space, respectively, which is consistent with the aim of LDA that tries to maximizing the within-class similarity and minimizing the between-class similarity simultaneously. In this paper, the trace ratio form is used.

For classification problems, after obtaining the transformation matrix W, the data sample x in the original feature space can be transformed into the discriminant space by computing xW. Then, nearest neighbor classifier can be used to make predictions in the discriminant space.

3. MULTI-TASK MULTI-VIEW DISCRIMINANT ANALYSIS

For Multi-task Multi-View (MTMV) learning, if we do not consider the relationships between multiple views and multiple tasks, a naive way to use LDA is as follows. For each view v in each task t, as described in Section 2.1, we can compute its corresponding between-class scatter matrix $S_{t,b}^v$ and total scatter matrix $S_{t,h}^v$. Using the trace ratio form, the optimization problem can be formulated as:

$$W_t^{v*} = \arg\max_{W_t^{v\top} W_t^v = I_l} \frac{\text{tr}(W_t^{v\top} S_{t,b}^v W_t^v)}{\text{tr}(W_t^{v\top} S_{t,h}^v W_t^v)}, \quad (3)$$

where l is the reduced dimensionality of the trace ratio form, $W_t^v \in \mathbb{R}^{d_v\times l}$ is the transformation matrix for dimensionality reduction for view v in task t, and d_v is the number of features in view v.

When multiple tasks with multiple views are available, it is better to share knowledge among them to obtain improved results compared to learning them separately. Instead of directly sharing some class-dependent model parameters among multiple tasks, we propose a multi-task multi-view discriminant analysis (MAMUDA) method to take advantage of the common structure representing some characteristics of the application. By solving the optimization problem, the transformation matrices can be obtained and a simple nearest neighbor classifier can be used to perform classification in the transformed lower-dimensional discriminant space.

3.1 Knowledge Shared among Multiple Tasks and Multiple Views

When multiple related tasks are available, it is better to learn them together and share some knowledge among them to get better results. So, based on Eq. (3), combining all the tasks and all the views' optimization problems into a unified form, we have the following optimization problem:

$$\max_{\{W_t^v\}, W_t^{v\top} W_t^v = I_l} \frac{\text{tr}(\sum_{t=1}^T \sum_{v=1}^V W_t^{v\top} S_{t,b}^v W_t^v)}{\text{tr}(\sum_{t=1}^T \sum_{v=1}^V W_t^{v\top} S_{t,h}^v W_t^v)} \quad (4)$$

The purpose of the optimization problem is to find the transformation matrices $\{W_t^v\}$ that transform the data from the original feature space into a discriminant space, where the data become more separable. Further, to facilitate the information sharing among multiple tasks, the transformation process is divided into two steps: (a) the data from all the tasks are transformed from their corresponding original feature space into a common intermediate latent semantic space, which reflects the intrinsic characteristics of the applications; (b) data from each view of each task are transformed from the common intermediate latent space into their corresponding discriminant space. Along this line, the optimization problem can be explicitly described as:

$$\max_{Q_t^v, R_t^v} \frac{\text{tr}(\sum_{t=1}^T \sum_{v=1}^V R_t^{v\top} Q_t^{v\top} S_{t,b}^v Q_t^v R_t^v)}{\text{tr}(\sum_{t=1}^T \sum_{v=1}^V R_t^{v\top} Q_t^{v\top} S_{t,h}^v Q_t^v R_t^v)} \quad (5)$$

where

- $Q_t^v R_t^v = W_t^v \in \mathbb{R}^{d_v\times l}$ is the overall transformation matrix;
- $Q_t^v \in \mathbb{R}^{d_v\times \bar{l}}$ ($Q_t^{v\top} Q_t^v = I_{\bar{l}}$) is the transformation matrix that transforms the v-th view data in task t from the original feature space into an intermediate latent semantic space, this intermediate space is shared by all the views in every task, $\bar{l}$ is the dimension of the intermediate latent space;
- $R_t^v \in \mathbb{R}^{\bar{l}\times l}$ ($R_t^{v\top} R_t^v = I_l$) is the transformation matrix that transforms the data from view v in task t from the common intermediate latent space into their corresponding discriminant space.

$\{Q_t^v\}$ are dependent on the views and tasks. Through matrices $\{Q_t^v\}$, the original data from all the views in every task, which have different feature representations, are transformed into a common intermediate latent semantic space. In addition, since the multiple tasks are from the same application, there will exist a common structure that is shared by all the tasks representing some characteristics of the application itself. The common structure can be used to facilitate the learning process of $\{R_t^v\}$, by taking advantage of the common intermediate latent space. Meanwhile, each task will have some specific characteristics that are only contained in this task. So, multiple views of this task can be used to jointly learn these task-specific structures. As a result, the transformation matrix R_t^v can be divided into two parts: one part is for the common structure shared by multiple tasks corresponding to the common discriminant components, while the other part learns the task-specific discriminant components that is shared by different views of this task. Different tasks and different views can share

knowledge through the common structures. The optimization problem can be described as:

$$\max_{Q_t^v, R, R_t} \frac{\mathrm{tr}(\sum_{t=1}^{T}\sum_{v=1}^{V}[R,R_t]^\top Q_t^{v\top} S_{t,b}^v Q_t^v [R,R_t])}{\mathrm{tr}(\sum_{t=1}^{T}\sum_{v=1}^{V}[R,R_t]^\top Q_t^{v\top} S_{t,h}^v Q_t^v [R,R_t])} \quad (6)$$
$$\text{s.t.} \quad Q_t^{v\top} Q_t^v = I_{\bar{l}}, R^\top R = I_{l'}, {R_t}^\top R_t = I_{l-l'}$$

where

- $R_t^v = [R, R_t] \in \mathbb{R}^{\bar{l}\times l}$ is the transformation matrix as described above;
- $R \in \mathbb{R}^{\bar{l}\times l'}$ is the common transformation matrix shared by all the views from every task, they can share knowledge through this matrix;
- $R_t \in \mathbb{R}^{\bar{l}\times(l-l')}$ is the task-specific transformation matrix, which is shared by all the views in task t, different views in task t share knowledge through this common matrix;

It must be noted that the number l' of shared discriminant components can be adjusted to accommodate to the relatedness of different tasks. For closely related tasks, we may use a large l' even equal to the total discriminant dimension l, and for loosely related tasks, we may use a small l' even equal to zero which is equivalent to no knowledge shared between different tasks.

3.2 Classif cation in Discriminant Space

When the transformation matrices $\{Q_t^v\}$ and $\{R_t^v\}$ are obtained, it is very straightforward to transform the data samples from the original feature space into low-dimensional representation in the discriminant space. Indeed, for each task, different views of features are available, and we can obtain the new representation for each view. For the i-th data sample with view v in task t, denoted by $x_{t,i}^v$, its new representation is $x_{t,i}^v Q_t^v R_t^v$. $R_t^v = [R, R_t]$ are the same for different views in a specific task, which means different views are transformed into a same discriminant space. Each view's data may have noises. So, average different views' representations for each task and obtain the final representation is a better choice. The final representation can be written as $\frac{1}{V}\sum_{v=1}^{V} x_{t,i}^v Q_t^v R_t^v$. Then, the nearest neighbor classifier can be used to make predictions in the discriminant feature space.

3.3 Optimization Procedure

It is difficult to solve the optimization problem (6) with respect to $\{Q_t^v\}$, $\{R_t\}$ and R jointly. In the following, an alternating optimization algorithm is presented. Specifically, we optimize the objective function with respect to each variable while the other variables are fixed. This procedure is repeated until convergence.

3.3.1 Computation of R with Fixed $\{Q_t^v\}$ and $\{R_t\}$

Note that $\mathrm{tr}([R,R_t]^\top Q_t^{v\top} S_{t,b}^v Q_t^v [R,R_t]) = \mathrm{tr}(R^\top Q_t^{v\top} S_{t,b}^v Q_t^v R) + \mathrm{tr}({R_t}^\top Q_t^{v\top} S_{t,b}^v Q_t^v R_t)$. When $\{Q_t^v\}$ and $\{R_t\}$ are fixed, the optimization problem (6) becomes:

$$\max_R \frac{\mathrm{tr}\left(R^\top(\sum_{t=1}^{T}\sum_{v=1}^{V} Q_t^{v\top} S_{t,b}^v Q_t^v)R\right) + al'}{\mathrm{tr}\left(R^\top(\sum_{t=1}^{T}\sum_{v=1}^{V} Q_t^{v\top} S_{t,h}^v Q_t^v)R\right) + bl'} \quad (7)$$

where a and b are constants:

$$\begin{aligned} a = &\ \tfrac{1}{l'}\mathrm{tr}(\textstyle\sum_{t=1}^{T}\sum_{v=1}^{V} {R_t}^\top Q_t^{v\top} S_{t,b}^v Q_t^v R_t) \\ b = &\ \tfrac{1}{l'}\mathrm{tr}(\textstyle\sum_{t=1}^{T}\sum_{v=1}^{V} {R_t}^\top Q_t^{v\top} S_{t,h}^v Q_t^v R_t) \end{aligned} \quad (8)$$

According to the constraints in Eq.(6), $R^\top R = I_{l'}$, so $\mathrm{tr}(R^\top R) = l'$. So Eq.(7) can be rewritten as:

$$\max_R \frac{\mathrm{tr}\left(R^\top \bar{S}_b R\right)}{\mathrm{tr}\left(R^\top \bar{S}_h R\right)} \quad \text{s.t.} \quad R^\top R = I_{l'} \quad (9)$$

where

$$\begin{aligned} \bar{S}_b = &\ \textstyle\sum_{t=1}^{T}\sum_{v=1}^{V} Q_t^{v\top} S_{t,b}^v Q_t^v + aI_{\bar{l}} \\ \bar{S}_h = &\ \textstyle\sum_{t=1}^{T}\sum_{v=1}^{V} Q_t^{v\top} S_{t,h}^v Q_t^v + bI_{\bar{l}} \end{aligned} \quad (10)$$

The problem in Eq.(9) has the same formulation as the trace ratio form of LDA, so a similar iterative method as in [23] is given in Algorithm 1 to solve it. As proved in [23], this algorithm will converge and return the globally optimal solution.

Algorithm 1 Computation of R with Fixed $\{Q_t^v\}$ and $\{R_t\}$

Input: $\bar{S}_b$, $\bar{S}_h$
Output: R
Method:
1: Initialize $R^{(0)}$ that satisfies ${R^{(0)}}^\top R^{(0)} = I_{l'}$;
2: **for** $k = 1$ to $maxIteNum$ **do**
3: Compute the trace ratio value as:
$$\lambda^k = \mathrm{tr}({R^{(k-1)}}^\top \bar{S}_b R^{(k-1)})/\mathrm{tr}({R^{(k-1)}}^\top \bar{S}_h R^{(k-1)})$$
4: Construct the trace difference problem as:
$$R^{(k)} = \arg\max_{R^\top R = I_{l'}} \mathrm{tr}[R^\top(\bar{S}_b - \lambda^k \bar{S}_h)R]$$
5: Solve the trace difference problem using the eigenvalue decomposition method, $R^{(k)}$ contains the top l' eigenvectors of $(\bar{S}_b - \lambda^k \bar{S}_h)$;
6: Let $S_{tmp} = R^{(k)}{R^{(k)}}^\top \bar{S}_h R^{(k)}{R^{(k)}}^\top$;
7: Let $R^{(k)}$ be the eigenvector matrix of S_{tmp} corresponding to the top l' eigenvalues;
8: If $\|R^{(k)} - R^{(k-1)}\|_F \le \varepsilon$, then break;
9: **end for**
10: **return** $R = R^{(k)}$.

3.3.2 Computation of R_t with Fixed R, $\{Q_t^v\}$ and $\{R_i\}(i \neq t)$

When R, $\{Q_t^v\}$ and $\{R_i\}(i \neq t)$ are fixed, the optimization problem (6) becomes:

$$\max_{R_t} \frac{\mathrm{tr}\left({R_t}^\top(\sum_{v=1}^{V} Q_t^{v\top} S_{t,b}^v Q_t^v)R_t\right) + a_t(l-l')}{\mathrm{tr}\left({R_t}^\top(\sum_{v=1}^{V} Q_t^{v\top} S_{t,h}^v Q_t^v)R_t\right) + b_t(l-l')} \quad (11)$$

where a_t and b_t are constants:

$$\begin{aligned} a_t = &\ \tfrac{1}{l-l'}\Big(\mathrm{tr}(\textstyle\sum_{i=1}^{T}\sum_{v=1}^{V} R^\top Q_i^{v\top} S_{i,b}^v Q_i^v R) \\ &+ \mathrm{tr}(\textstyle\sum_{i=1,i\neq t}^{T}\sum_{v=1}^{V} {R_i}^\top Q_i^{v\top} S_{i,b}^v Q_i^v R_i)\Big) \\ b_t = &\ \tfrac{1}{l-l'}\Big(\mathrm{tr}(\textstyle\sum_{i=1}^{T}\sum_{v=1}^{V} R^\top Q_i^{v\top} S_{i,h}^v Q_i^v R) \\ &+ \mathrm{tr}(\textstyle\sum_{i=1,i\neq t}^{T}\sum_{v=1}^{V} {R_i}^\top Q_i^{v\top} S_{i,h}^v Q_i^v R_i)\Big) \end{aligned} \quad (12)$$

According to the constraints in Eq.(6), ${R_t}^\top R_t = I_{l-l'}$, so $\mathrm{tr}({R_t}^\top R_t) = l - l'$. Then, Eq.(11) can be rewritten as:

$$\max_{R_t} \frac{\mathrm{tr}({R_t}^\top \bar{S}_{t,b} R_t)}{\mathrm{tr}({R_t}^\top \bar{S}_{t,h} R_t)} \quad \text{s.t.} \quad {R_t}^\top R_t = I_{l-l'} \tag{13}$$

where

$$\begin{aligned} \bar{S}_{t,b} = & \sum_{v=1}^{V} {Q_t^v}^\top S_{t,b}^v Q_t^v + a_t I_{\bar{l}} \\ \bar{S}_{t,h} = & \sum_{v=1}^{V} {Q_t^v}^\top S_{t,h}^v Q_t^v + b_t I_{\bar{l}} \end{aligned} \tag{14}$$

The problem in Eq.(13) has the same formulation as the problem in Eq.(9), so it can be solved by a very similar iterative method as in Algorithm 1. Also, the globally optimal solution can be obtained.

3.3.3 Computation of Q_t^v with Fixed R, $\{R_t\}$ and $\{Q_i^j\}(i \neq t, j \neq v)$

When R, $\{R_t\}$ and $\{Q_i^j\}(i \neq t, j \neq v)$ are fixed, and $R_t^v = [R, R_t]$, the optimization problem in Eq. (6) becomes:

$$\max_{Q_t^v} \frac{\mathrm{tr}({R_t^v}^\top {Q_t^v}^\top S_{t,b}^v Q_t^v R_t^v) + a_t^v l}{\mathrm{tr}({R_t^v}^\top {Q_t^v}^\top S_{t,h}^v Q_t^v R_t^v) + b_t^v l} \tag{15}$$

where a_t^v and b_t^v are constants:

$$\begin{aligned} a_t^v = & \ \tfrac{1}{l}\mathrm{tr}(\textstyle\sum_{(i,j)\neq(t,v)} {R_i^j}^\top {Q_i^j}^\top S_{i,b}^j Q_i^j R_i^j) \\ b_t^v = & \ \tfrac{1}{l}\mathrm{tr}(\textstyle\sum_{(i,j)\neq(t,v)} {R_i^j}^\top {Q_i^j}^\top S_{i,h}^j Q_i^j R_i^j) \end{aligned} \tag{16}$$

According to the constraints in Eq. (6), ${Q_t^v}^\top Q_t^v = I_{\bar{l}}$ and ${R_t^v}^\top R_t^v = I_l$, then ${R_t^v}^\top {Q_t^v}^\top Q_t^v R_t^v = {R_t^v}^\top R_t^v = I_l$, so $\mathrm{tr}({R_t^v}^\top {Q_t^v}^\top Q_t^v R_t^v) = l$. The optimization problem in Eq. (15) can be rewritten as:

$$\max_{Q_t^v} \frac{\mathrm{tr}({R_t^v}^\top {Q_t^v}^\top \bar{S}_{t,b}^v Q_t^v R_t^v)}{\mathrm{tr}({R_t^v}^\top {Q_t^v}^\top \bar{S}_{t,h}^v Q_t^v R_t^v)} \tag{17}$$

where

$$\bar{S}_{t,b}^v = S_{t,b}^v + a_t^v I_{d_v}, \quad \bar{S}_{t,h}^v = S_{t,h}^v + b_t^v I_{d_v}. \tag{18}$$

We further rewrite it as:

$$\max_{Q_t^v} \frac{\mathrm{tr}({Q_t^v}^\top \bar{S}_{t,b}^v Q_t^v A_t^v)}{\mathrm{tr}({Q_t^v}^\top \bar{S}_{t,h}^v Q_t^v A_t^v)} \quad \text{s.t.} \quad {Q_t^v}^\top Q_t^v = I_{\bar{l}} \tag{19}$$

where $A_t^v = R_t^v {R_t^v}^\top$. This problem has the same formulation as Eq.(7) in [27], the optimization procedure of it is given in Algorithm 2.

The solution of the trace difference problem in Step 5 of Algorithm 2 is given by the Theorem 1 [27]. The algorithm will return the globally optimal solution of problem (19).

THEOREM 1. *Let A be a real $p \times p$ symmetric matrix and B be a real $q \times q$ positive semidefinite matrix where $p > q$. Then*

$$\max_{W \in \mathbb{R}^{p\times q}, W^\top W = I_q} \mathrm{tr}(W^\top A W B) = \sum_{i=1}^{q} \lambda_i(A)\lambda_i(B),$$

where $\lambda_i(A)$ denotes the i-th largest eigenvalue of matrix A. The optimal solution satisfies $W^* = U_a {U_b}^\top Q$, where U_a is the eigenvector matrix of A corresponding to the top q eigenvalues, U_b is the eigenvector matrix of B, and Q is any $q \times q$ orthogonal matrix.

Algorithm 2 Computation of Q_t^v with Fixed R, $\{R_t\}$ and $\{Q_i^j\}_{(i,j)\neq(t,v)}$

Input: $\bar{S}_{t,b}^v$, $\bar{S}_{t,h}^v$ and A_t^v
Output: Q_t^v
Method:
1: Initialize $Q_t^{v(0)}$ that satisfies ${Q_t^{v(0)}}^\top Q_t^{v(0)} = I_{\bar{l}}$;
2: **for** $k = 1$ to $maxIteNum$ **do**
3: Compute the trace ratio value as:

$$\lambda^k = \frac{\mathrm{tr}({Q_t^{v(k-1)}}^\top \bar{S}_{t,b}^v Q_t^{v(k-1)} A_t^v)}{\mathrm{tr}({Q_t^{v(k-1)}}^\top \bar{S}_{t,h}^v Q_t^{v(k-1)} A_t^v)}$$

4: Construct the trace difference problem as:

$$Q_t^{v(k)} = \arg \max_{{Q_t^v}^\top Q_t^v = I_{\bar{l}}} \mathrm{tr}[{Q_t^v}^\top (\bar{S}_{t,b}^v - \lambda_k \bar{S}_{t,h}^v) Q_t^v A_t^v]$$

5: Solve the trace difference problem: Let $Q_t^{v(k)} = U_s {U_a}^\top$, where U_s is the eigenvector matrix contains the top $\bar{l}$ eigenvectors of $(\bar{S}_{t,b}^v - \lambda_k \bar{S}_{t,h}^v)$ and U_a is the eigenvector matrix of A_t^v;
6: Let $S_{tmp} = Q_t^{v(k)} {Q_t^{v(k)}}^\top \bar{S}_{t,h}^v Q_t^{v(k)} {Q_t^{v(k)}}^\top$;
7: Let $Q_t^{v(k)}$ be the eigenvector matrix of S_{tmp} corresponding to the top $\bar{l}$ eigenvalues;
8: If $\|Q_t^{v(k)} - Q_t^{v(k-1)}\|_F \leq \varepsilon$, then break;
9: **end for**
10: **return** $Q_t^v = Q_t^{v(k)}$.

3.4 Dealing with Missing View Data

In many real world problems, multiple tasks do not have the same set of views. Some tasks may miss some views of features that are existing in other tasks. It is straightforward to deal with the problems that have structured missing views using the methods described in the above sections. If view v is missing from task t, then the variables concerning this view in this task will be eliminated in the computation process. It is obvious that the structured missing views do not affect the formulation of calculation formula of other variables, thus the proposed optimization problem in Eq.(6) can be regarded as a general framework for dealing with MTMV problems.

4. EXPERIMENTAL RESULTS

In this section, we systematically evaluate the effectiveness of the proposed Multi-tAsk MUlti-view Discriminant Analysis (MAMUDA) method. The classification results for problems with both complete views and missing views are given to show the performances of MAMUDA.

4.1 Data Preparation

Two applications have multiple tasks with multiple views are considered in this paper, some statistics of them are summarized in Table 1, where T1 represents the first task in a problem, V1, V2 and V3 denotes three different views of features. Originally, every task in the two applications has all the views of features. We also consider the missing view problems by randomly eliminating some views. Missing views are shown in Table 1 using the symbol '×' and existing views using '√'. It can be seen that different tasks in a problem have different number of classes, i.e., they do not share the same set of class labels. Thus, they are MTMV

problems with heterogeneous tasks. The specific characteristics of these two problems are given below.

Leaves: The leaves data set [11] includes leaves from one hundred plant species that are divided into 32 different genuses, and 16 samples of leaves for each plant species are presented. For each sample, three views of features are available, including shape descriptor, fine scale margin and texture histogram, and each view has 64 features. 4 genuses that have 4 or more plant species are selected to form 4 tasks, and the aim of the problem is to discriminate different species in a genus, which is a multi-class classification problem. Overall, the problem has 4 tasks with 3 views, and different tasks have different number of classes. Some views are randomly eliminated from the data to form its corresponding missing view problem, as shown in Table 1.

Face: In the face recognition problem, we use 3 face databases: PIE [19], Yale[3] and ORL [3]. PIE contains facial images for 68 persons. We choose the Pose C09 from PIE, which contains 24 images for each person. Yale contains 165 images for 15 individuals. There are 11 images for each individual, and each one with different facial expression or configuration. ORL contains 400 face images of 40 persons, each having 10 images. These face images have significant variations in pose and scale. Before the experiment, each image is converted to gray scale and normalized to two different sizes of 32×32 pixels and 28×28 pixels, which provides two different views of features for the image. The face recognition problem for each database can be seen as a task, so there are 3 tasks with 2 views in total. Each task corresponds to a multi-class classification problem where the number of classes in each task is equal to the number of persons in each database. After randomly eliminating some views, the missing view problem is shown in Table 1.

Table 1: Some Statistics of the Problems

Problem	T	V1	V2	V3	class #	sample #
Leaves	T1	√	√	√	11	176
	T2	√	√	√	5	80
	T3	√	√	√	4	64
	T4	√	√	√	38	608
Leaves (Missing View)	T1	×	√	√	11	176
	T2	√	×	√	5	80
	T3	√	√	×	4	64
	T4	√	√	√	38	608
Face	T1	√	√	NA	68	1632
	T2	√	√	NA	15	165
	T3	√	√	NA	40	400
Face (Missing View)	T1	√	×	NA	68	1632
	T2	√	√	NA	15	165
	T3	√	×	NA	40	400

4.2 Experimental Settings

4.2.1 Benchmark Algorithms

Since most previous methods [14, 26, 16] assume that multiple tasks in a problem should be similar and share the same set of class labels, and they are formulated for binary classification problems, they cannot be directly applied to the application scenarios in Section 4.1. Therefore, we first compare MAMUDA with several methods that can solve the MTMV problems with heterogeneous tasks. For all these algorithms, they learn discriminant space representations and a simple nearest neighbor is used to perform classification in the lower-dimensional space. These algorithms are described as follows:

- **LDA:** Linear Discriminant Analysis (LDA) algorithm for single task with single view [23]. Each view in each task is computed separately using LDA, and there is no knowledge shared among multiple tasks and views. For each view in each task, we can obtain a new representation in discriminant space, and a classification result can be obtained for each of them.
- **MTDA:** Multi-task Discriminant Analysis (MTDA) [27] is a single view multi-task learning algorithm. Here, the MTMV problem is divided into several MTL problems, each for a view, and MTDA is used to solve each of the MTL problems.
- **MVDA:** Single-task Multi-view Discriminant Analysis (MVDA) method can be seen as a special case of the MAMUDA method. Here, the MTMV problem is divided into several MVL problems, one for each task. MVDA is used to solve each of the MVL problems. Because all the views of features for a task are transformed into the same discriminant space, the final representation of the task can be obtained by averaging all the views' representations.
- **MAMUDA-s:** A simplified version of MAMUDA, by setting $l' = l$ in Eq.(6), which means that there are only common structures shared by multiple tasks and there are no task-specific structure. This is designed to show the benefit of using task-specific structures.

Second, previous MTMV algorithms are designed for binary classification problems. To further compare our algorithm with the previous MTMV algorithms, we need to convert the multi-class classification problem into a set of binary classification problems using the one-against-all method for previous algorithms. Two MTMV algorithms are used:

- **IteM²:** It is a transductive MTMV algorithm for binary classification problems [14].
- **regMVMT:** It is an inductive binary classification algorithm, which assumes all tasks should be similar to achieve good performance [26].

4.2.2 Evaluation Metrics

In the experiments, different numbers of samples are randomly selected for each class to investigate the effect of varying the size of training set on classification performances. For each configuration, we perform 10 random trials and the average error rates are reported. The average error rate is the percentage of wrongly classified samples among all the test samples from all tasks.

4.3 Learning Results for Problems with Complete Views

For the Leaves problem, we randomly select $n \in \{2, 3, 4, 5\}$ samples for each class as training set and the rest as the test set. Here we do not set $n = 1$, since in this case there is only one labeled sample in each class, the between-class scatter matrix is equal to the total scatter matrix and we cannot build a meaningful optimization problem. The parameters

[3]http://cvc.yale.edu/projects/yalefaces/yalefaces.html

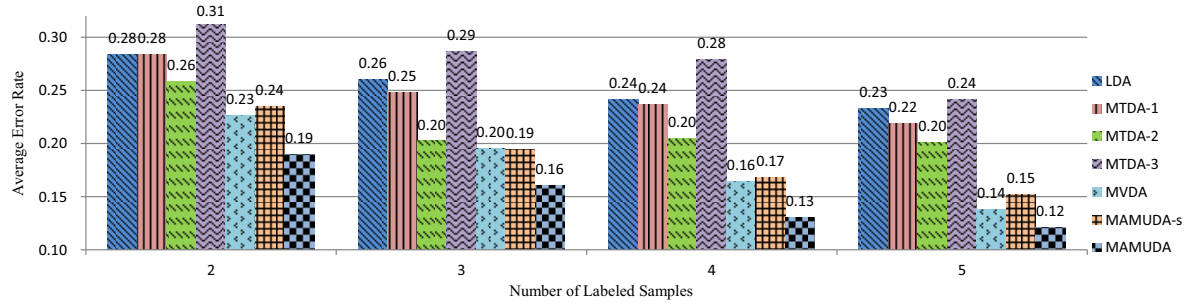


Figure 4: Experimental Results for the Leaves Problem

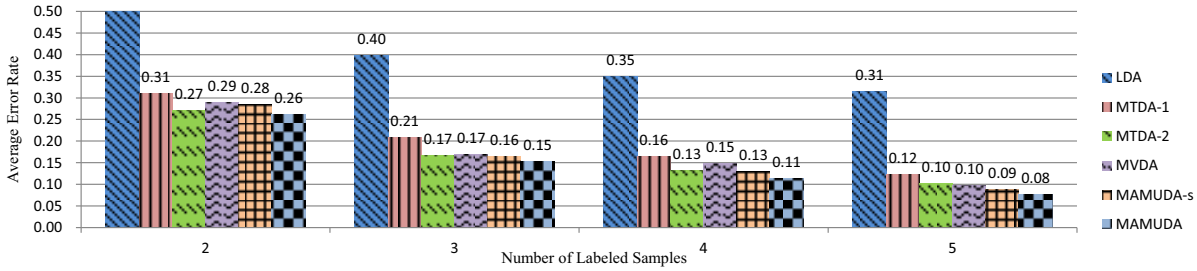


Figure 5: Experimental Results for the Face Problem

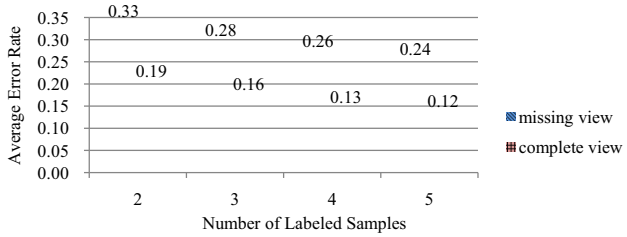
used are: $\bar{l} = 50$, $l = 100$ and $l' = 50$. Each experiment is repeated 10 times, and the average error rates of all the tasks are shown in Figure 4, where for LDA algorithm the average error rates of the three views from multiple tasks are given. For MTDA algorithm, each view's results are given separately, and MTDA-i represents the results of MTDA algorithm using the i-th view data. It can be seen that there are great distinctions on different views's ability for classification using MTDA algorithm. MAMUDA obtains the best results for all the different numbers of training samples. LDA algorithm does not perform well as it does not share knowledge among multiple tasks or multiple views. MAMUDA is better than MTDA and MVDA, which demonstrates MAMUDA can benefit from sharing knowledge among both multiple tasks and multiple views. MAMUDA is also better than MAMUDA-s algorithm, which shows that MAMUDA can take advantage of the task-specific structures to obtain additional improvement.

For the Face problem, we also randomly select $n \in \{2, 3, 4, 5\}$ samples for each class as training set and the rest as the test set. The parameters used are: $\bar{l} = 300$, $l = 300$ and $l' = 200$. Each experiment is repeated 10 times, the average error rates of all the tasks are shown in Figure 5. The results are very similar to those of Leaves. LDA algorithm is the worst among all the algorithms. MAMUDA obtains the best results for all the different numbers of training samples. MAMUDA is also better than MAMUDA-s algorithm, which does not consider the task-specific information.

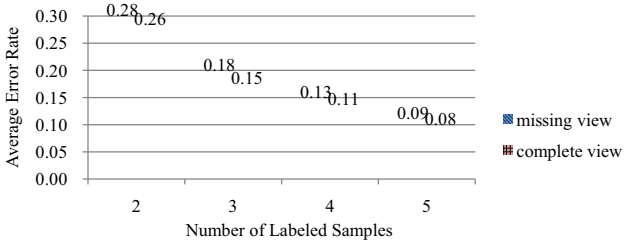
4.4 Learning Results for Problems with Missing Views

For both Leaves and Face problems with missing views, as shown in Table 1, we randomly select $n \in \{2, 3, 4, 5\}$ samples for each class as the training data and the rest as the test data. Each experiment is repeated 10 times, the average error rates of all the tasks are shown in Figure 6 and 7. Again, MAMUDA obtains the best results for all the different numbers of training samples. This shows that MAMUDA can solve the problems with missing views.

We also compared the results of MAMUDA algorithm for problems with missing views and complete views, which are shown in Figure 8. It can be seen that, for both Leaves Problem and Face problem, when some views are missing, the results become worse than the problems with complete views. So, MAMUDA can take advantage of multiple views and leverage them to obtain good results.



(a) *Leaves Problem*



(b) *Face Problem*

Figure 8: Comparison of Results from Missing Views and Complete Views Problems

4.5 A Comparison with existing MTMV algorithms

Existing MTMV algorithms [14, 26] are designed for binary classification problems. To compare with them, we need to convert the multi-class problems into binary classification problems. As each task in our problems is a multi-class problem, for each class, a new binary classification task is constructed according to the one-against-all method. Thus, task 1 in the Leaves problems is converted into 11 bi-

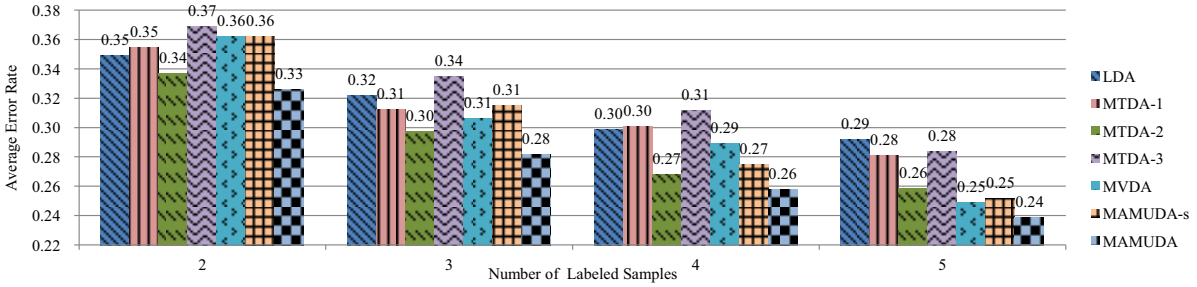


Figure 6: Experimental Results for the Leaves Problem with Missing Views

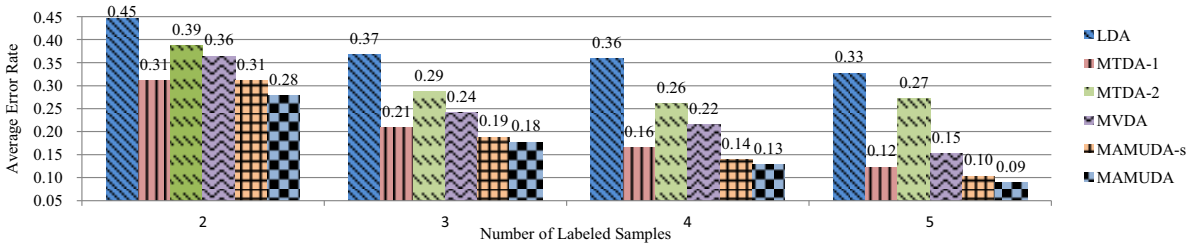


Figure 7: Experimental Results for the Face Problem with Missing Views

nary tasks and the Leaves problem contains 58 binary classification tasks in total. The average error rate for each algorithm is shown in Table 2. It must be noted that, for regMVMT and IteM^2 algorithms, the results shown in the table are the error rates of the transformed binary classification problems. The results need to be mapped to the results of multi-class problems, which will become much worse as the binary results is already too poor. It is observed that, MAMUDA substantially outperforms existing MTMV algorithms. Similarly, using this method, the Face problem can be converted into a new problem with 123 binary classification tasks. Unfortunately, regMVMT algorithm [26] cannot solve this problem due to this algorithm's high time complexity and space complexity[4]. Thus only IteM^2 is compared with our algorithm, which also does not obtain acceptable results for our heterogeneous problems.

Table 2: Classification Error Rates for Previous MTMV Algorithms (Reesults of the Transformed Binary Problem)

training sample #		2	3	4	5
Leaves	regMVMT	0.51	0.49	0.46	0.54
	IteM^2	0.44	0.43	0.40	0.40
Face	IteM^2	0.50	0.49	0.47	0.46

It can be seen that MAMUDA achieves the best results, and both the state-of-the-art MTMV algorithms can not perform well for our problems. We conjecture that there are two reasons: (1) They need to transform the multi-class problem into a number of binary classification problems, which may lead to the class-unbalance and degrade the performance. (2) They do not consider the special requirement that the tasks are heterogeneous. Indeed, they are not very suitable for the problems considered in this paper, as they assume all the tasks are similar.

[4] In fact, the authors of regMVMT have explicitly stated in their paper that "*Due to the limit of the matrix size in most computer systems, our algorithm can only handle up to tens of tasks, and learning problems with hundreds of tasks or more are hence beyond the scope of this paper.*"

4.6 Parameter Sensitivity

To check the effect of varying the number of dimensions $\bar{l}$ of the common intermediate latent space, the total number of discriminant dimensions l and the number of shared discriminant dimensions l' in Eq.(6), we vary them in a wide range of values.

First, for Leaves data set, we vary $\bar{l}$ from 10 to 60, and correspondingly set $l = \bar{l}$ and $l' = \frac{1}{2}\bar{l}$, the results are given in Fig.9(a). We can find that when the value of $\bar{l}$ is larger than 50, the results are stable. So the default value of $\bar{l}$ is set as 50 for Leaves problem. Similarly, for Face data set, we vary $\bar{l}$ from 200 to 600, the results are shown in Fig.9(b). It can be seen that the results are good when $\bar{l}$ are not too small or too large. So, $\bar{l}$ need to be tuned by the users to obtain good result. Also, this parameter is not difficult to be set, as the results are good in a wide range of parameter settings.

Then, while $\bar{l}$ are fixed with the best parameter, we vary l from 10 to 100 and correspondingly set $l' = \frac{1}{2}l$. The results for Leaves problem are given in Fig.9(c). Similarly, the results for Face problem with different l values are shown in Fig.9(d). It can be observed that, for both the two data sets, the results are relatively stable with not too small values of l. Therefore, the default value of l for Leaves problem is set as 100 and 300 for Face problem.

Finally, we fix $\bar{l} = 50$ and $l = 100$, and vary l' from 0 to 50 for Leaves problem. The results are shown in Fig.9(e). Similarly, while other parameters are fixed, Face problem's results are in Fig.9(f). For both of the two problems, the results are stable with not too small l' values. So, we set $l' = 50$ for Leaves problem and $l' = 200$ for Face problem.

5. RELATED WORKS

Generally speaking, related works can be grouped into the following three categories.

The first category includes the studies of Multi-task learning (MTL), which conducts multiple related learning tasks simultaneously so that the label information in one task can be used for other tasks. The earliest MTL method

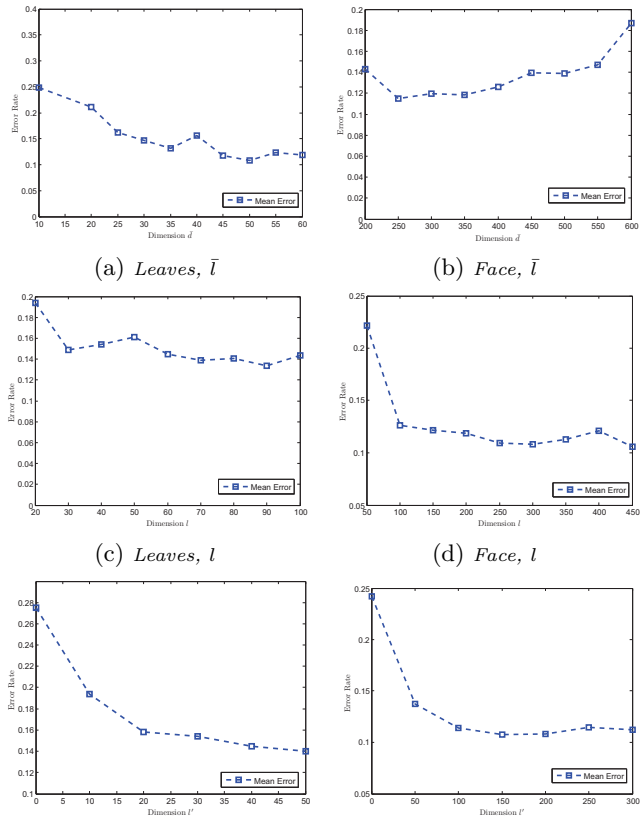


Figure 9: Prediction Accuracy with respect to Different Dimension Settings

[5] learns a shared hidden layer representation for different tasks. Multi-task feature learning learns a low-dimensional representation which is shared across a set of multiple related tasks [2, 15]. The methods to learn predictive structures on hypothesis spaces from multiple learning tasks are also proposed [1, 6]. These methods can learn some shared features or shared structures between different task, but they based on the original feature and can only use information from the same view data, which means they cannot share information among different views. Supposing that all the tasks are similar, a regularization formulation is proposed for MTL [9]. MTL can be modeled by stochastic process methods, such as [22, 25]. To deal with outlier tasks, a robust multi-task learning algorithm is proposed [7]. These methods share knowledge by placing a common prior on the model parameters of each task in hierarchical Bayesian models and explicitly share some model parameters or model structure among tasks, which requires that multiple tasks in the problem are similar and have the same set of class labels. So, they cannot be used for problems with heterogeneous tasks. MTDA algorithm [27] is a single view multi-task learning algorithm that can deal with learning tasks with different data representations. It can be seen as a special case of our MAMUDA method, which does not share information among different views of a same task. Also, MTDA only uses the shared structure among tasks and does not consider the task-specific structures.

The second category includes the works on Multi-view Learning (MVL). The basic idea of MVL is to make use of the consistency among different views to achieve better performance. One of the earliest works on multi-view learning is co-training algorithm [4], which uses one view's predictor to enlarge the training set for other views. Nigam and Ghani compared co-training, EM and co-EM methods, and showed that co-EM algorithm is the best among the three approaches [18]. Some improvements of co-training algorithm are also proposed [17, 24]. Other methods are based on co-regularization framework. Sindhwani et al. [20] proposed a learning framework for multi-view regularization. SVM-2K [10] is a method which uses kernels for two views learning. Sindhwani and Rosenberg [21] constructed a single Reproducing Kernel Hilbert Spaces (RKHSs) with a data-dependent "co-regularization" norm that reduces MVL to standard supervised learning. Chen et al. [8] presented a large-margin learning framework to discover a predictive latent subspace representation shared by multiple views. All these methods are designed for single task learning.

Finally, the third category includes the efforts on multi-task multi-view (MTMV) learning with homogeneous tasks. He and Lawrence [14] proposed a graph-based framework which takes full advantage of information among multiple tasks and multiple views, and an iterative algorithm (IteM^2) was developed to optimize the model. However, multiple tasks share information by directly sharing some class-dependent model parameters. This requires multiple tasks have the same class label set and the classes in different tasks have an one-to-one relationship. So, it can not solve problems with heterogeneous tasks. Also, it can only deal with problems with nonnegative feature values. regMVMT [26] uses co-regularization to obtain functions that are consistent with each other on the unlabeled samples for different views. Across different tasks, additional regularization functions are utilized to ensure the learned functions are similar for multiple tasks, which implies that it can not solve problems with heterogeneous tasks. CSL-MTMV [16] is a shared structure learning framework, which can learn shared predictive structures on common views from multiple related tasks, and use the consistency among different views to improve the performance. Also, it is a binary classification algorithm and is not convenient for multi-class problems.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we formulated a new type of multi-task multi-view (MTMV) learning problem, i.e., MTMV with heterogeneous tasks, and a Multi-tAsk MUlti-view Discriminant Analysis (MAMUDA) method is proposed to solve it. In MAMUDA, both the shared structure that represents the characteristics of the application and the task-specific structures can be combined into a unified formulation to facilitate the learning process. Furthermore, an alternating optimization algorithm is developed to solve the problem. Finally, experiments on several real world problems, with both complete views and missing views, demonstrate the effectiveness of the proposed method.

To solve the MTMV problem, we ingeniously extended LDA methods to the multi-task multi-view setting, and the framework is very general and useful. It is worth mentioning that many other dimension reduction models (e.g., canonical correlation analysis model [13]) can also be optional for our general framework, which will be further investigated in the future.

7. ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (No. 61175052, 61203297, 61035003, 61473273, 61473274), National High-tech R&D Program of China (863 Program) (No. 2014AA012205, 2013AA01A606, 2012AA011003).

8. REFERENCES

[1] R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:01, 2005.

[2] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *Advances in Neural Information Processing Systems*, pages 41 – 48, Vancouver, BC, Canada, 2007.

[3] P. Belhumeur, J. Hespanha, and D. Kriegman. Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, JUL 1997.

[4] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, COLT' 98, pages 92–100, New York, NY, USA, 1998. ACM.

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

[6] J. Chen, L. Tang, J. Liu, and J. Ye. A convex formulation for learning shared structures from multiple tasks. In *Proceedings of the 26th International Conference On Machine Learning, ICML 2009*, pages 137 – 144, Montreal, QC, Canada, 2009.

[7] J. Chen, J. Zhou, and J. Ye. Integrating low-rank and group-sparse structures for robust multi-task learning. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 42 – 50, San Diego, CA, United states, 2011.

[8] N. Chen, J. Zhu, and E. P. Xing. Predictive subspace learning for multi-view data: A large margin approach. In *Annual Conference on Neural Information Processing Systems 2010, NIPS 2010*, Vancouver, BC, Canada, 2010.

[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, WA, United states, 2004.

[10] J. D. Farquhar, D. R. Hardoon, H. Meng, J. Shawe-Taylor, and S. Szedmak. Two view learning: Svm-2k, theory and practice. In *Advances in Neural Information Processing Systems*, pages 355 – 362, Vancouver, BC, Canada, 2005.

[11] A. Frank and A. Asuncion. UCI machine learning repository, 2013.

[12] K. Fukunaga. *Introduction to statistical pattern recognition*. Academic press, 1990.

[13] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12):2639–2664, 2004.

[14] J. He and R. Lawrence. A graph-based framework for multi-task multi-view learning. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, pages 25 – 32, Bellevue, WA, United states, 2011.

[15] A. Jalali, P. Ravikumar, S. Sanghavi, and C. Ruan. A dirty model for multi-task learning. In *24th Annual Conference on Neural Information Processing Systems 2010, NIPS 2010*, Vancouver, BC, Canada, 2010.

[16] X. Jin, F. Zhuang, S. Wang, Q. He, and Z. Shi. Shared structure learning for multiple tasks with multiple views. In *Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2013)*, volume 8189 LNAI, pages 353 – 368, Prague, Czech republic, 2013.

[17] I. Muslea, S. Minton, and C. A. Knoblock. Active + Semi-supervised Learning = Robust Multi-View Learning. In *International Conference on Machine Learning*, pages 435–442, 2002.

[18] K. Nigam and R. Ghani. Analyzing the effectiveness and applicability of co-training. In *International Conference on Information and Knowledge Management*, pages 86–93, 2000.

[19] T. Sim, S. Baker, and M. Bsat. The CMU pose, illumination, and expression database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1615–1618, DEC 2003.

[20] V. Sindhwani, P. Niyogi, and M. Belkin. A Co-Regularization Approach to Semi-supervised Learning with Multiple Views. In *Workshop on Learning with Multiple Views, International Conference on Machine Learning*, 2005.

[21] V. Sindhwani and D. S. Rosenberg. An rkhs for multi-view learning and manifold co-regularization. In *ICML*, pages 976 – 983, Helsinki, Finland, 2008.

[22] G. Skolidis and G. Sanguinetti. Bayesian multitask classification with gaussian process priors. *IEEE Transactions on Neural Networks*, 22(12):2011–2021, 2011.

[23] H. Wang, S. Yan, D. Xu, X. Tang, and T. Huang. Trace ratio vs. ratio trace for dimensionality reduction. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Minneapolis, MN, United states, 2007.

[24] S. Yu, B. Krishnapuram, R. Rosales, and R. Bharat Rao. Bayesian co-training. *Journal of Machine Learning Research*, 12:2649 – 2680, 2011.

[25] S. Yu, V. Tresp, and K. Yu. Robust multi-task learning with t-processes. In *Twenty-Fourth International Conference on Machine Learning*, volume 227, pages 1103 – 1110, Corvalis, OR, United states, 2007.

[26] J. Zhang and J. Huan. Inductive multi-task learning with multiple view data. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 543 – 551, Beijing, China, 2012.

[27] Y. Zhang and D.-Y. Yeung. Multi-task learning in heterogeneous feature spaces. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 1, pages 574 – 579, San Francisco, CA, United states, 2011.