MACHINE LEARNING: SEVERAL ADVANCES IN LINEAR DISCRIMINANT ANALYSIS, MULTI-VIEW REGRESSION AND SUPPORT VECTOR MACHINE

by

SHUAI ZHENG

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2017

This dissertation is dedicated to my parents, Shuqin An and Fushun Zheng.

ACKNOWLEDGEMENTS

I would like to thank my supervising professor Dr. Chris Ding for constantly motivating and encouraging me, and for his invaluable advice during the course of my doctoral studies. I also would like to thank Dr. Heng Huang, Dr. Junzhou Huang and Dr. Jeff Lei for their interest in my research, for taking time to serve in my dissertation committee and for their comments, suggestions and guidance.

I appreciate those valuable discussions, brain storms with other group members during this work. I was fortunate enough to have so many great friends along the way and I do appreciate those days and nights we spent together. I also received a lot support from advisers and staffs at Department of Computer Science and Engineering. I am very grateful for those help.

My parents have been a life-long source of love, encouragement and inspiration to me throughout both good and difficult times. Without their support, this work could never have been possible. To them, I owe everything.

April 3rd, 2017

ABSTRACT

MACHINE LEARNING: SEVERAL ADVANCES IN LINEAR DISCRIMINANT ANALYSIS, MULTI-VIEW REGRESSION AND SUPPORT VECTOR MACHINE

SHUAI ZHENG, Ph.D.

The University of Texas at Arlington, 2017

Supervising Professor: Chris Ding

Machine learning technology is now widely used in engineering, science, finance, healthcare, etc. In this dissertation, we make several advances in machine learning technologies for high dimensional data analysis, image data classification, recommender systems and classification algorithms.

In this big data era, many data are high dimensional data which is difficult to analyze. We propose two efficient Linear Discriminant Analysis (LDA) based methods to reduce data to low dimensions. Kernel alignment measures the degree of similarity between two kernels. We propose *kernel alignment inspired LDA* to find a subspace to maximize the alignment between subspace-transformed data kernel and class indicator kernel. Classical LDA uses arithmetic mean of all between-class distances. However, arithmetic mean between-class distance has some limitations. First, large between-class distance could dominate the arithmetic mean. Second, arithmetic mean does not consider pairwise between-class distance and thus some classes may overlap with each other in the subspace. We propose *harmonic mean based LDA* to overcome the limitations of classical LDA.

Low-rank models can capture the correlations between data. We propose an efficient low-rank regression model for image and website classification and a regularized Singular Value Decomposition (SVD) model for recommender system. Real life data often includes information from different channels. These different aspects/channels of the same object are called multi-view data. In this work, we propose a *multi-view low-rank regression model* by imposing low-rank constraints on multi-view data and we provide a closed-form solution to the multi-view low-rank regression model. Recommender system is very important for online advertising, online shopping, social network, etc. In recent applications, regularization becomes an increasing trend. We present a *regularized SVD (RSVD) model for recommender system* to improve standard SVD based models.

Support Vector Machine (SVM) is an efficient classification approach, which finds a hyperplane to separate data from different classes. This hyperplane is determined by support vectors. In existing SVM formulations, the objective function uses L2 norm or L1 norm on slack variables. The number of support vectors is a measure of generalization errors. In this work, we propose a *Minimal SVM*, which uses L0.5 norm on slack variables. The result model further reduces the number of support vectors and increases the classification performance.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Machine learning technology is now widely used in engineering, science, finance, healthcare, etc. For example, self-driving car applied machine learning technology to navigate and detect objects in videos and images; online advertising business needs recommender system technology to attract most number of web clicks and boost online transactions. All these applications need reliable and accurate machine learning models. In this dissertation, we make several advances in machine learning technologies for high dimensional data analysis, image data classification, recommender systems and classification algorithms.

High dimensional data is very common nowadays. For example, a photo taken by a smarter phone has about 12 million pixels. If we represent this photo using pixel vectors, the vector will have dimension of 12 million. High dimensional data is very difficult to analyze and it takes a lot of storage and computing resources to process high dimensional data. Linear Discriminant Analysis (LDA) is an efficient dimensionality reduction algorithm. We propose two efficient LDA based methods to reduce data to low dimensions. Kernel alignment measures the degree of similarity between two kernels. We propose *kernel alignment inspired LDA* (kaLDA) to find a subspace to maximize the alignment between subspace-transformed data kernel and class indicator kernel. Classical LDA uses arithmetic mean of all between-class distances. However, arithmetic mean between-class distance has some limitations. First, large between-class distance could dominate the arithmetic mean. Second, arithmetic mean does not consider pairwise between-class distance and thus some

classes may overlap with each other in the subspace. We propose *harmonic mean based LDA* (HLDA) to overcome the limitations of classical LDA. We conducted extensive experiments using kaLDA and HLDA on image data and found that classification accuracy can be improved using kaLDA and HLDA.

Low-rank models can capture the correlations between data. We propose an efficient low-rank regression model for image and website classification. Real life data often includes information from different channels. These different aspects/channels of the same object are called multi-view data. For images, multi-view data can be different features extracted from the same image, such as HOG, SIFT and GIST features. For website, multi-view data can be word content, images, and hyperlinks in the same webpage. In this work, we propose a *multi-view low-rank regression model* by imposing low-rank constraints on multi-view data and we provide a closed-form solution to the multi-view low-rank regression model. Results on real life image and website data show the proposed multi-view low-rank regression model can improve classification accuracy efficiently.

Recommender system is very important for online advertising, online shopping, social network, etc. Singular Value Decomposition (SVD) is widely used in recommender system by exploring correlations between users and correlations between items. In recent applications, regularization becomes an increasing trend. We present a *regularized SVD (RSVD) model for recommender system* and a closed form solution to RSVD to improve standard SVD based models. Experiments on movie rating and joke rating data show that recommendations using RSVD is more accurate.

Support Vector Machine (SVM) is an efficient classification approach, which finds a hyperplane to separate data from different classes. This hyperplane is determined by support vectors. In existing SVM formulations, the objective function uses L_2 norm or L_1 norm on slack variables. The number of support vectors is a measure

of generalization errors. In this work, we propose a *Minimal SVM*, which uses $L_{0.5}$ norm on slack variables. Experiments on image data show that the Minimal SVM model further reduces the number of support vectors and increases the classification performance.

This dissertation is organized as follows: Chapter 2 introduces kernel alignment inspired LDA; Chapter 3 introduces harmonic mean based LDA; Chapter 4 introduces multi-view low-rank regression model; Chapter 5 introduces regularized SVD (RSVD) model for recommender system; Chapter 6 introduces Minimal SVM; Chapter 7 concludes this dissertation.

CHAPTER 2

KERNEL ALIGNMENT INSPIRED LINEAR DISCRIMINANT ANALYSIS

2.1 Introduction

Kernel alignment [1] is a way to incorporate class label information into kernels which are traditionally directly constructed from data without using class labels. Kernel alignment can be viewed as a measurement of consistency between the similarity function (the kernel) and class structure in the data. Improving this consistency helps to enforce data become more separated when using the class label aligned kernel. Kernel alignment has been applied to pattern recognition and feature selection recently [2, 3, 4, 5, 6].

In this chapter, we find that if we use the widely used linear kernel and a kernel built from class indicators, the resulting kernel alignment function is very similar to the widely used linear discriminant analysis (LDA), using the well-known between-class scatter matrix S_b and total scatter matrix S_t. We call this objective function as kernel alignment induced LDA (kaLDA) [7]. If we transform data into a linear subspace, the optimal solution is to maximize this kaLDA.

We further analyze this kaLDA and propose a Stiefel-manifold gradient descent algorithm to solve it. We also extend kaLDA to multi-label problems. Surprisingly, the scatter matrices arising in multi-label kernel alignment are identical those matrices developed in Multi-label LDA [8].

We perform extensive experiments by comparing kaLDA with other approaches on 8 single-label datasets and 6 multi-label data sets. Results show that kernel align-

ment LDA approach has good performance in terms of classification accuracy and F1 score.

2.2 From Kernel Alignment to LDA

Kernel Alignment is a similarity measurement between a kernel function and a target function. In other words, kernel alignment evaluates the degree of fitness between the data in kernel space and the target function. For this reason, we usually set the target function to be the class indicator function. The other kernel function is the data matrix. By measuring the similarity between data kernel and class indicator kernel, we can get a sense of how easily this data can be separated in kernel subspace. The alignment of two kernels $\mathcal{K}_1$ and $\mathcal{K}_2$ is given as [1]:

$$A(\mathcal{K}_1, \mathcal{K}_2) = \frac{\mathrm{Tr}(\mathcal{K}_1\mathcal{K}_2)}{\sqrt{\mathrm{Tr}(\mathcal{K}_1\mathcal{K}_1)}\sqrt{\mathrm{Tr}(\mathcal{K}_2\mathcal{K}_2)}}. \tag{2.1}$$

We first introduce some notations, and then present Theorem 1 and kernel alignment projective function.

Let data matrix be $X \in \mathbb{R}^{p\times n}$ and $X = (\mathbf{x}_1, \cdots, \mathbf{x}_n)$, where p is data dimension, n is number of data points, $\mathbf{x}_i$ is a data point. Let normalized class indicator matrix be $Y \in \mathbb{R}^{n\times K}$, which was used to prove the equivalence between PCA and K-means clustering [9, 10], and

$$Y_{ik} = \begin{cases} \frac{1}{\sqrt{n_k}}, & \text{if point } i \text{ is in class } k. \\ 0, & \text{otherwise.} \end{cases} \tag{2.2}$$

where K is total class number, n_k is the number of data points in class k. Class mean is $\mathbf{m}_k = \sum_{\mathbf{x}_i\in k} \mathbf{x}_i/n_k$ and total mean of data is $\mathbf{m} = \sum_i \mathbf{x}_i/n$.

Theorem 1. *Define data kernel $\mathcal{K}_1$ and class label kernel $\mathcal{K}_2$ as follows:*

$$\mathcal{K}_1 = X^TX, \;\; \mathcal{K}_2 = YY^T, \tag{2.3}$$

we have

$$A(\mathcal{K}_1, \mathcal{K}_2) = c\frac{TrS_b}{\sqrt{TrS_t^2}} \tag{2.4}$$

where $c = 1/\sqrt{Tr(YY^T)^2}$ *is a constant independent of* X.

Furthermore, let $G \in \mathbb{R}^{p\times k}$ *be a linear transformation to a* k*-dimensional subspace*

$$\widetilde{X} = G^T X, \;\; \widetilde{\mathcal{K}}_1 = \widetilde{X}^T \widetilde{X}, \tag{2.5}$$

we have

$$A(\widetilde{\mathcal{K}}_1, \mathcal{K}_2) = c\frac{Tr(G^T S_b G)}{\sqrt{Tr(G^T S_t G)^2}} \tag{2.6}$$

where

$$S_b = \sum_{k=1}^{K} n_k(\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T, \tag{2.7}$$

$$S_t = \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^T, \tag{2.8}$$

Theorem 1 shows that kernel alignment can be expressed using scatter matrices S_b and S_t. In applications, we adjust G such that kernel alignment is maximized, i.e., we solve the following problem:

$$\max_G \frac{\mathrm{Tr}(G^T S_b G)}{\sqrt{\mathrm{Tr}(G^T S_t G)^2}}. \tag{2.9}$$

In general, columns of G are assumed to be linearly independent.

A striking feature of this kernel alignment problem is that it is very similar to classic LDA.

2.2.1 Proof of Theorem 1 and Analysis

Here we note a useful lemma and then prove Theorem 1.

In most data analysis, data are centered, i.e., $\sum_i \mathbf{x}_i = \mathbf{0}$. Here we assume data is already centered. The following results remain correct if data is not centered. We have the following relations:

Lemma 1. *Scatter matrices S_b, S_t can be expressed as:*

$$S_b = XYY^TX^T, \tag{2.10}$$

$$S_t = XX^T. \tag{2.11}$$

These results are previously known, for example, Theorem 3 of [10].

Proof of Theorem 1. To prove Eq.(2.4), we substitute $\mathcal{K}_1, \mathcal{K}_2$ into Eq.(2.1) and obtain, noting $\mathrm{Tr}(AB) = \mathrm{Tr}(BA)$,

$$A(\mathcal{K}_1, \mathcal{K}_2) = \frac{\mathrm{Tr}(XYY^TX^T)}{\sqrt{\mathrm{Tr}(XX^T)^2}\sqrt{\mathrm{Tr}(YY^T)^2}} = c\frac{\mathrm{Tr}S_b}{\sqrt{\mathrm{Tr}S_t^2}}.$$

where we used Lemma 1. $c = 1/\sqrt{\mathrm{Tr}(YY^T)^2}$ is a constant independent of data X.

To prove Eq.(2.6),

$$A(\widetilde{\mathcal{K}}_1, \mathcal{K}_2) = c\frac{\mathrm{Tr}(G^TXYY^TX^TG)}{\sqrt{\mathrm{Tr}(G^TXX^TG)^2}} = c\frac{\mathrm{Tr}(G^TS_bG)}{\sqrt{\mathrm{Tr}(G^TS_tG)^2}},$$

thus we obtain Eq.(2.6) using Lemma 1.

2.2.2 Relation to Classical LDA

In classical LDA, the between-class scatter matrix S_b is defined as Eq.(2.7), and the within-class scatter matrix S_w and total scatter matrix S_t are defined as:

$$S_w = \sum_{k=1}^{K}\sum_{\mathbf{x}_i \in k}(\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)^T,\ S_t = S_b + S_w, \tag{2.12}$$

where $\mathbf{m}_k$ and $\mathbf{m}$ are class means. Classical LDA finds a projection matrix $G \in \mathbb{R}^{p\times(K-1)}$ that minimizes S_w and maximizes S_b using the following objective:

$$\max_G \mathrm{Tr}\frac{G^TS_bG}{G^TS_wG}, \tag{2.13}$$

or

$$\max_G \frac{\mathrm{Tr}(G^T S_b G)}{\mathrm{Tr}(G^T S_w G)}. \tag{2.14}$$

Eq.(2.14) is also called trace ratio (TR) problem [11]. It is easy to see [1] that Eq.(2.14) can be expressed as

$$\max_G \frac{\mathrm{Tr}(G^T S_b G)}{\mathrm{Tr}(G^T S_t G)}. \tag{2.15}$$

As we can see, kernel alignment LDA objective function Eq.(2.9) is very similar to Eq.(2.15). Thus kernel alignment provides an interesting alternative explanation of LDA. In fact, we can similarly show that in Eq.(2.9), S_w is also maximized as in the standard LDA. First, Eq.(2.9) is equivalent to

$$\max_G \mathrm{Tr}(G^T S_b G) \;\; s.t. \;\; \mathrm{Tr}(G^T S_t G)^2 = \eta,$$

where η is a fixed-value. The precise value of η is unimportant, since the scale of G is undefined in LDA: if G^* is an optimal solution, and r is any real number, $G^{**} = rG^*$ is also an optimal solution with the same optimal objective function value. The above optimization is approximately equivalent to

$$\max_G \mathrm{Tr}(G^T S_b G) \;\; s.t. \;\; \mathrm{Tr}(G^T S_t G) = \eta,$$

This is same as

$$\max_G \mathrm{Tr}(G^T S_b G) \;\; s.t. \;\; \mathrm{Tr}(G^T S_w G) = \eta - \mathrm{Tr}(G^T S_b G),$$

In other words, S_b is maximized while S_w is minimized — recovering the LDA main theme.

[1] Eq.(2.14) is equivalent to $\min \frac{\mathrm{Tr}(G^T S_w G)}{\mathrm{Tr}(G^T S_b G)}$, which is $\min \left(\frac{\mathrm{Tr}(G^T S_w G)}{\mathrm{Tr}(G^T S_b G)} + 1 \right)$. Reversing to maximization and using $S_t = S_b + S_w$, we obtain Eq.(2.15).

2.3 Computational Algorithm

In this section, we develop efficient algorithm to solve kaLDA objective function Eq.(2.9):

$$\max_G J_1 = \frac{\mathrm{Tr}(G^T S_b G)}{\sqrt{\mathrm{Tr}(G^T S_t G)^2}}, \quad s.t. \;\; G^T G = I. \tag{2.16}$$

The condition $G^T G = I$ ensures different columns of G mutually independent. The gradient of $J_1(G)$ is

$$\nabla J_1 \triangleq \frac{\partial J_1}{\partial G} = 2\frac{A}{\sqrt{\mathrm{Tr}D^2}} - 2\frac{\mathrm{Tr}B}{(\mathrm{Tr}D^2)^{\frac{3}{2}}}CD, \tag{2.17}$$

where $A = S_b G, \; B = G^T A, \; C = S_t G, \; D = G^T C$.

Constraint $G^T G = I$ enforces G on the Stiefel manifold. Variations of G on this manifold is parallel transport, which gives some restriction to the gradient. This has been been worked out in [12]. The gradient that reserves the manifold structure is

$$\nabla J_1 - G[\nabla J_1]^T G. \tag{2.18}$$

Thus the algorithm computes the new G is given as follows:

$$G \leftarrow G - \eta(\nabla J_1 - G[\nabla J_1]^T G). \tag{2.19}$$

The step size η is usually chosen as:

$$\eta = \tau \|G\|_1 / \|\nabla J_1 - G(\nabla J_1)^T G\|_1, \; \tau = 0.001 \sim 0.01. \tag{2.20}$$

where $\|G\|_1 = \sum_{ij} |G_{ij}|$.

Occasionally, due to the loss of numerical accuracy, we use projection $G \leftarrow G(G^T G)^{-\frac{1}{2}}$ to restore $G^T G = I$. Starting with the standard LDA solution of G, this algorithm is iterated until the algorithm converges to a local optimal solution. In fact, objective function will converge quickly when choosing η properly. Figure 2.1

Algorithm 1 $[G] = kaLDA(X, Y)$

Input: Data matrix $X \in \mathbb{R}^{p \times n}$, class indicator matrix $Y \in \mathbb{R}^{n \times K}$

Output: Projection matrix $G \in \mathbb{R}^{p \times k}$

1: Compute S_b and S_t using Eq.(2.10) and Eq.(2.11)

2: Initialize G using classical LDA solution

3: **repeat**

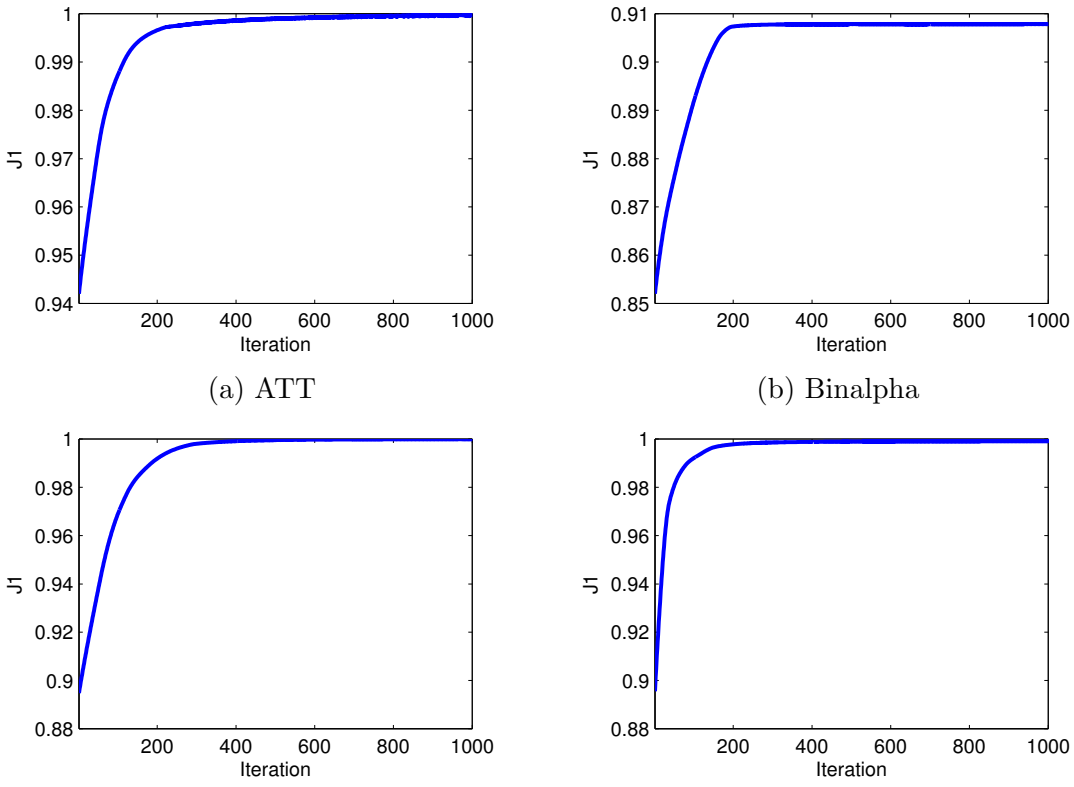
4: Compute gradient using Eq.(2.17)

5: Update G using Eq.(2.19)

6: **until** J_1 Converges

shows that J_1 converges in about 200 iterations when $\tau = 0.001$, for datasets ATT, Binalpha, Mnist, and Umist (more details about the datasets will be introduced in experiment section). In summary, kernel alignment LDA (kaLDA) procedure is shown in Algorithm 1.

To show the effectiveness of proposed kaLDA, we visualize a real dataset in 2-D subspace in Figure 2.2. In this example, we take 3 classes of 644-dimension Umist data, 18 data points in each class. Figure 2.2a shows the original data projected in 2-D PCA subspace. Blue points are in class 1; red circle points are in class 2; black square points are in class 3. Data points from the three classes are mixed together in 2-D PCA subspace. It is difficult to find a linear boundary to separate points of different classes. Figure 2.2b shows the data in 2-D standard LDA subspace. We can see that data points in different classes have been projected into different clusters. Figure 2.2c shows the data projected in 2-D kaLDA subspace. Compared to Figure 2.2b, the within-class distance in Figure 2.2c is much smaller. The distance between different classes is larger.



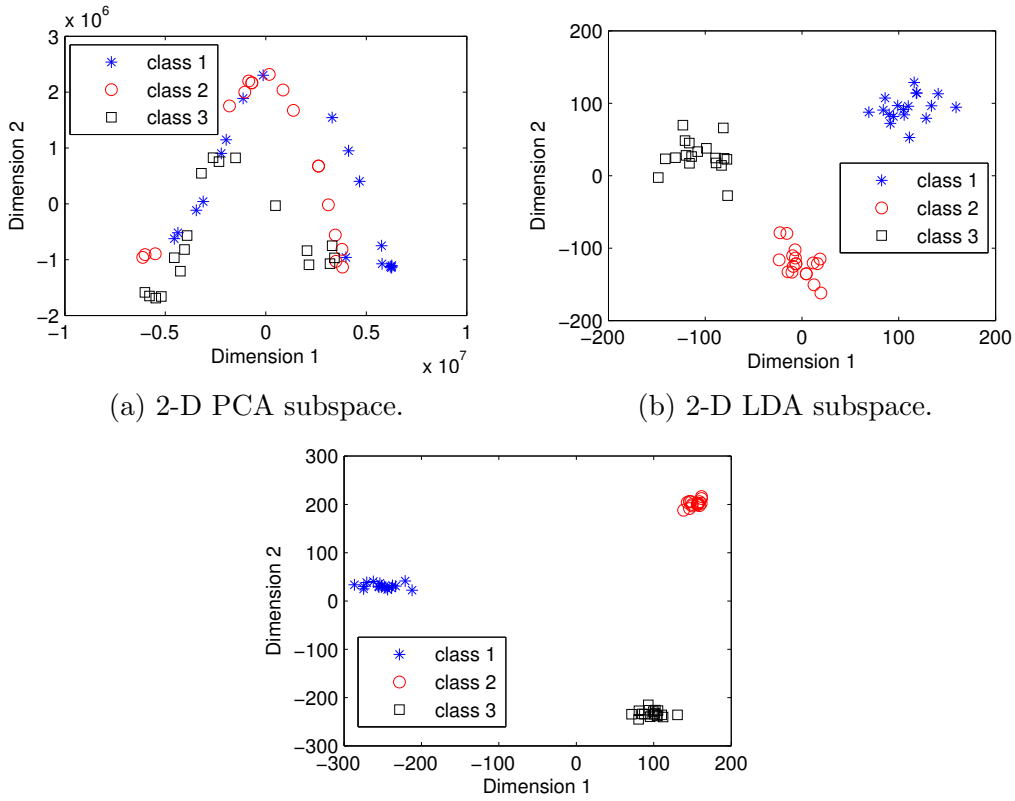
(a) ATT (b) Binalpha

(c) Mnist (d) Umist

Figure 2.1: Objective J1 converges using Stiefel-manifold gradient descent algorithm ($\tau = 0.001$).

2.4 Extension to Multi-label Data

Multi-label problem arises frequently in image and video annotations, multi-topic text categorization, music classification. etc.[8]. In multi-label data, a data point could have several class labels (belonging to several classes). For example, an image could have "cloud", "building", "tree" labels. This is different from the case of single-label problem, where one point can have only one class label. Multi-label is very natural and common in our everyday life. For example, a film can be simultaneously classified as "drama", "romance", "historic" (if it is about a true story). A news article can have topic labels such as "economics", "sports", etc.



(a) 2-D PCA subspace.

(b) 2-D LDA subspace.

(c) 2-D kaLDA subspace.

Figure 2.2: Visualization of Umist data in 2-D PCA, 2-D LDA and 2-D kaLDA subspace.

Kernel alignment approach can be easily and naturally extended to multi-label data, because the class label kernel can be clearly and unambiguously defined using class label matrix Z on both single label and multi-label data sets. The data kernel is defined as usual. In the following we further develop this approach.

One important result of our kernel alignment approach for single label data is that it has close relationship with LDA. For multi-label data, each data point could belong to several classes. The standard scatter matrices S_b, S_w are ambiguous, because S_b, S_w are only defined for single label data where each data point belongs to one class only. However, our kernel alignment approach on multi-label data leads to new definitions of scatter matrices and similar objective function; this can be viewed

as the generalization of LDA from single-label data to multi-label data via kernel alignment approach.

Indeed, the new scatter matrices we obtained from kernel alignment approach are identical to the so-called "multi-label LDA" [8] developed from a class-separate, probabilistic point of view, very different from our point of view. The fact that these two approaches lead to the same set of scatter matrices show that the resulting multi-label LDA framework has a broad theoretical basis.

We first present some notations for multi-label data and then describe the kernel alignment approach for multi-label data in Theorem 2. The class label matrix $Z \in \mathbb{R}^{n\times K}$ for data $X \in \mathbb{R}^{p\times n}$ is given as:

$$Z_{ik} = \begin{cases} 1, & \text{if point } i \text{ is in class } k. \\ 0, & \text{otherwise.} \end{cases} \tag{2.21}$$

Let $\tilde{n}_k = \sum_{i=1}^{n} Z_{ik}$ be the number of data points in class k. Note that for multi-label data, $\sum_{k=1}^{K} \tilde{n}_k > n$. The **normalized** class indicator matrix $\widetilde{Y} \in \mathbb{R}^{n\times K}$ is given as:

$$\widetilde{Y}_{ik} = \begin{cases} \frac{1}{\sqrt{\tilde{n}_k}}, & \text{if point } i \text{ is in class } k. \\ 0, & \text{otherwise.} \end{cases} \tag{2.22}$$

Let $\rho_i = \sum_{k=1}^{K} Z_{ik}$ be the number of classes that $\mathbf{x}_i$ belongs to. Thus ρ_i are the weights of $\mathbf{x}_i$. Define the diagonal weight matrix $\Omega = \text{diag}(\rho_1, \cdots, \rho_n)$. The kernel alignment formulation for multi-label data can be stated as

Theorem 2. *For multi-label data X, let the data kernel and class label kernel be*

$$\mathcal{K}_1 = \Omega^{\frac{1}{2}} X^T X \Omega^{\frac{1}{2}}, \;\; \mathcal{K}_2 = \Omega^{-\frac{1}{2}} \widetilde{Y}\widetilde{Y}^T \Omega^{-\frac{1}{2}}. \tag{2.23}$$

We have the alignment

$$A(\mathcal{K}_1, \mathcal{K}_2) = c\frac{TrS_b}{\sqrt{TrS_t^2}} \tag{2.24}$$

where $c = 1/\sqrt{Tr(\Omega^{-1}\widetilde{Y}\widetilde{Y}^T)^2}$ *is a constant independent of data* X*, and* S_b*,* S_t *are given in Eqs.(2.27, 2.28).*

Furthermore, let $G \in \mathbb{R}^{p\times k}$ *be the linear transformation to a* k*-dimensional subspace,*

$$\widetilde{X} = G^T X, \;\; \widetilde{\mathcal{K}}_1 = \Omega^{1/2}\widetilde{X}^T\widetilde{X}\Omega^{1/2}, \tag{2.25}$$

we have

$$A(\widetilde{\mathcal{K}}_1, \mathcal{K}_2) = c\frac{Tr(G^T S_b G)}{\sqrt{Tr(G^T S_t G)^2}} \tag{2.26}$$

The matrices S_b, S_t in Theorem 2 are defined as:

$$S_b = \sum_{k=1}^{K} \tilde{n}_k(\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T, \tag{2.27}$$

$$S_t = \sum_{k=1}^{K}\sum_{i=1}^{n} Z_{ik}(\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^T, \tag{2.28}$$

where $\mathbf{m}_k$ is the mean of class k and $\mathbf{m}$ is global mean, defined as:

$$\mathbf{m}_k = \frac{\sum_{i=1}^n Z_{ik}\mathbf{x}_i}{\tilde{n}_k}, \quad \mathbf{m} = \frac{\sum_{i=1}^n \rho_i \mathbf{x}_i}{\sum_{k=1}^K \tilde{n}_k}. \tag{2.29}$$

Therefore, we can seek an optimal subspace for multi-label data by solving Eq.(2.16) with S_b, S_t given in Eqs.(2.27,2.28)

2.4.1 Proof of Theorem 2 and Equivalence to Multi-label LDA

Here we note a useful lemma for multi-label data and then prove Theorem 2. We consider the case the data is centered, i.e., $\sum_{i=1}^n \rho_i \mathbf{x}_i = \mathbf{0}$. The results also hold when data is not centered, but the proofs are slightly complicated.

Lemma 2. *For multi-label data,* S_b, S_t *of Eqs.(2.27,2.28) can be expressed as*

$$S_b = X\widetilde{Y}\widetilde{Y}^T X^T \tag{2.30}$$

$$S_t = X\Omega X^T \tag{2.31}$$

Proof. From the definition of $\mathbf{m}_k$ and $\widetilde{Y}$ in multi-label data, we have

$$X\widetilde{Y} = (\mathbf{m}_1, \cdots, \mathbf{m}_K) \begin{pmatrix} \sqrt{\tilde{n}_1} & & \\ & \ddots & \\ & & \sqrt{\tilde{n}_K} \end{pmatrix}.$$

Thus $X\widetilde{Y}\widetilde{Y}^T X^T = \sum_{k=1}^{K} \tilde{n}_k \mathbf{m}_k \mathbf{m}_k^T$ recovers S_b of Eq.(2.27).

To prove Eq.(2.31), note that $X\Omega = (\rho_1 \mathbf{x}_1, \cdots, \rho_n \mathbf{x}_n)$, thus $X\Omega X^T = \sum_{i=1}^{n} \rho_i \mathbf{x}_i \mathbf{x}_i^T$.

□

Proof of Theorem 2. Using Lemma 2, to prove Eq.(2.24),

$$A(\mathcal{K}_1, \mathcal{K}_2) = c\frac{\mathrm{Tr}(X\widetilde{Y}\widetilde{Y}^T X^T)}{\sqrt{\mathrm{Tr}(X\Omega X^T)^2}} = c\frac{\mathrm{Tr}S_b}{\sqrt{\mathrm{Tr}S_t^2}},$$

where $c = 1/\sqrt{\mathrm{Tr}(\Omega^{-1}\widetilde{Y}\widetilde{Y}^T)^2}$ is independent of X.

To prove Eq.(2.26),

$$A(\widetilde{\mathcal{K}}_1, \mathcal{K}_2) = c\frac{\mathrm{Tr}(G^T X\widetilde{Y}\widetilde{Y}^T X^T G)}{\sqrt{\mathrm{Tr}(G^T X\Omega X^T G)^2}} = c\frac{\mathrm{Tr}(G^T S_b G)}{\sqrt{\mathrm{Tr}(G^T S_t G)^2}}.$$

For single-label data, $\rho_i = 1$, $\Omega = I$, $\tilde{n}_k = n_k$, Eqs.(2.30, 2.31) reduce to Eqs.(2.10, 2.11), and Theorem 2 reduces to Theorem 1.

As we can see, surprisingly, the scatter matrices S_b, S_t of Eqs.(2.27, 2.28) arising in Theorem 2 are identical to that in Multi-label LDA proposed in [8].

2.5 Related Work

Linear Discriminant Analysis (LDA) is a widely-used dimension reduction and subspace learning algorithm. There are many LDA reformulation publications in recent years. Trace Ratio problem is to find a subspace transformation matrix G such that the within-class distance is minimized and the between-class distance is maximized. Formally, Trace Ratio maximizes the ratio of two trace terms,

Table 2.1: Single-label datasets attributes.

Data	n	p	k
Caltec07	210	432	7
Caltec20	1230	432	20
MSRC	210	432	7
ATT	400	644	40
Binalpha	1014	320	26
Mnist	150	784	10
Umist	360	644	20
Pie	680	1024	68

Table 2.2: Classification accuracy on Single-label datasets ($K-1$ dimension).

Data	kaLDA	LDA	TR	sdpLDA	MMC	RLDA	OCM
Caltec07	0.7524	0.6619	0.6762	0.5619	0.6000	**0.7952**	0.7619
Caltec20	**0.7068**	0.6320	0.4465	0.3386	0.5838	0.6812	0.6696
MSRC	**0.7762**	0.6857	0.5714	0.5952	0.5667	0.7333	0.7286
ATT	**0.9775**	0.9750	0.9675	0.9750	0.9750	0.9675	0.9675
Binalpha	0.7817	0.6078	0.4620	0.2507	0.7638	0.7983	**0.8204**
Mnist	**0.8800**	0.8733	0.8667	0.8467	0.8467	0.8667	0.8467
Umist	0.9900	0.9900	**0.9917**	0.9133	0.9633	0.9800	0.9783
Pie	0.8765	**0.8838**	0.8441	0.8632	0.8676	0.6515	0.6515

$\max_G \mathrm{Tr}(G^T S_b G)/\mathrm{Tr}(G^T S_t G)$ [11, 13], where S_t is total scatter matrix and S_b is between-class scatter matrix. Other popular LDA approach includes, regularized LDA(RLDA) [14], Orthogonal Centroid Method (OCM) [15], Uncorrelated LDA (ULDA) [16], Orthogonal LDA (OLDA) [16], etc. These approaches mainly compute the eigendecomposition of matrix $S_t^{-1}S_b$, but use different formulation of total scatter matrix S_t [17].

Maximum Margin Criteria (MMC) [18] is a simpler and more efficient method. MMC finds a subspace projection matrix G to maximize $\mathrm{Tr}(G^T(S_b - S_w)G)$. Though in a different way, MMC also maximizes between-class distance while minimizing within-class distance. Semi-Definite Positive LDA (sdpLDA) [19] solves the maxi-

Table 2.3: Multi-label datasets attributes.

Data	n	p	k
MSRC-MOM	591	384	23
Barcelona	139	48	4
Emotion	593	72	6
Yeast	2,417	103	14
MSRC-SIFT	591	240	23
Scene	2,407	294	6

Table 2.4: Classification accuracy on Multi-label datasets ($K-1$ dimension).

Data	kaLDA	MLSI	MDDM	MLLS	MLDA
MSRC-MOM	**0.9150**	0.8962	0.9044	0.8994	0.9036
Barcelona	**0.6579**	0.6436	0.6470	0.6524	0.6290
Emotion	**0.7634**	0.7397	0.7540	0.7529	0.7619
Yeast	**0.7405**	0.7317	0.7371	0.7364	0.7368
MSRC-SIFT	0.8839	0.8762	0.8800	0.8807	**0.8858**
Scene	**0.8870**	0.8534	0.8713	0.8229	0.8771

mization of $\mathrm{Tr}(G^T(S_b - \lambda_1 S_w)G)$, where λ_1 is the largest eigenvalue of $S_w^{-1}S_b$. sdpLDA is derived from the maximum margin principle.

Multi-label problem arise frequently in image and video annotations and many other related applications, such as multi-topic text categorization [8]. There are many Multi-label dimension reduction approaches, such as Multi-label Linear Regression (MLR), Multi-label informed Latent Semantic Indexing (MLSI) [20], Multi-label Dimensionality reduction via Dependence Maximization (MDDM) [21], Multi-Label Least Square (MLLS) [22], Multi-label Linear Discriminant Analysis (MLDA) [8].

2.6 Experiments

In this section, we first compare kernel alignment LDA (kaLDA) with other six different methods on 8 single label data sets and compare kaLDA multi-label version with four other methods on 6 multi-label data sets.

2.6.1 Comparison with Trace Ratio w.r.t. subspace dimension

Eight single-label datasets are used in this experiment. These datasets come from different domains, such as image scene Caltec [23] and MSRC [24], face datasets ATT, Umist, Pie [25], and digit datasets Mnist [26] and Binalpha. Table 2.1 summarizes the attributes of those datasets.

Caltec07 and **Caltec20** are subsets of Caltech 101 data. Only the HOG feature is used in this chapter.

MSRC is a image scene data, includes tree, building, plane, cow, face, car and so on. It has 210 images from 7 classes and each image has 432 dimension.

ATT data contains 400 images of 40 persons, with 10 images for each person. The images has been resized to 28×23.

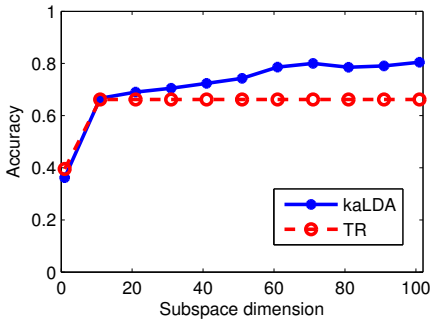
Binalpha data contains 26 binary hand-written alphabets. It has 1014 images in total and each image has 320 dimension.

Mnist is a handwritten digits dataset. The digits have been size-normalized and centred. It has 10 classes and 150 images in total, with 784 dimension each image.

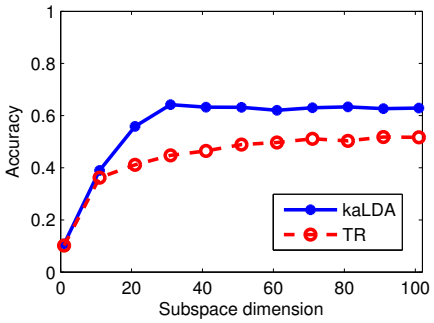
Umist is a face image dataset (Sheffield Face database) with 360 images from 20 individuals with mixed race, gender and appearance.

Pie is a face database collected by Carnegie Mellon Robotics Institute between October and December 2000. In total, it has 68 different persons.

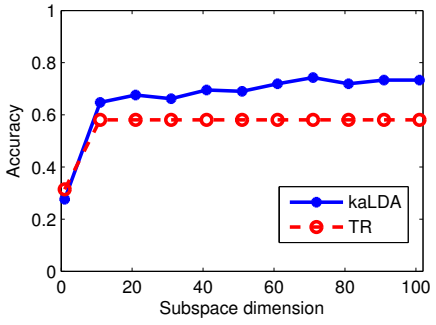
In this part, we compare the classification accuracy of kaLDA and Trace Ratio [11] with respect to subspace dimension. The dimension of the subspace that kaLDA can find is not restricted to $K-1$. After subspace projection, KNN classifier ($knn = 3$) is applied to perform classification. Results are shown in Figure 2.3. Solid line denotes kaLDA accuracy and dashed line denotes Trace Ratio accuracy. As we can see, in Figures 2.3a, 2.3b, 2.3c, 2.3g, and 2.3h, kaLDA has higher accuracy than Trace Ratio when using the same number of reduced features. In Figures 2.3d, 2.3e, 2.3f, kaLDA



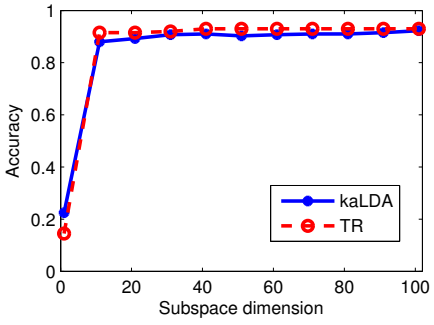
(a) Caltec07



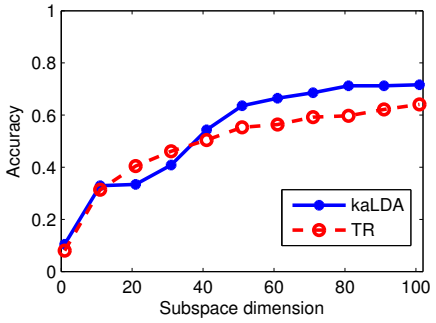
(b) Caltec20



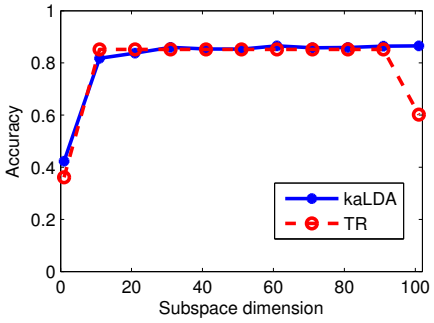
(c) MSRC



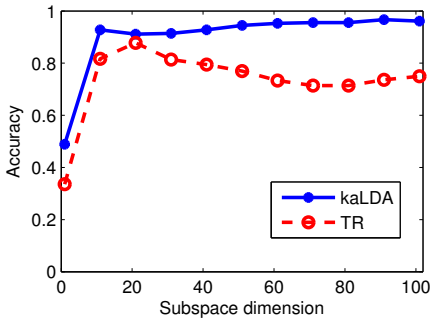
(d) ATT



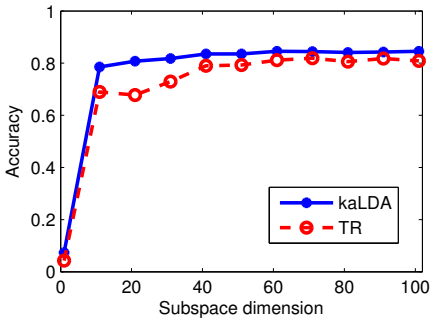
(e) Binalpha



(f) Mnist



(g) Umist



(h) Pie

Figure 2.3: Classification accuracy w.r.t. dimension of the subspace.

Table 2.5: Macro F1 score on Multi-label datasets ($K-1$ dimension).

Dataset	kaLDA	MLSI	MDDM	MLLS	MLDA
MSRC-MOM	**0.6104**	0.5244	0.5593	0.5426	0.5571
Barcelona	**0.7377**	0.7286	0.7301	0.7341	0.7169
Emotion	**0.6274**	0.5873	0.6101	0.6041	0.6200
Yeast	**0.5757**	0.5568	0.5696	0.5691	0.5693
MSRC-SIFT	0.4712	0.4334	0.4522	0.4544	**0.4773**
Scene	**0.6851**	0.5911	0.6411	0.5048	0.6568

Table 2.6: Micro F1 score on Multi-label datasets ($K-1$ dimension).

Dataset	kaLDA	MLSI	MDDM	MLLS	MLDA
MSRC-MOM	**0.5138**	0.4064	0.4432	0.4370	0.4448
Barcelona	**0.6969**	0.6891	0.6861	0.6904	0.6772
Emotion	**0.6203**	0.5779	0.6030	0.5961	0.6151
Yeast	**0.4249**	0.4026	0.4205	0.4216	0.4213
MSRC-SIFT	0.3943	0.3510	0.3637	0.3667	**0.3959**
Scene	**0.6966**	0.6006	0.6493	0.5062	0.6643

has competitive classification accuracy with Trace Ratio. However, kaLDA is more stable than Trace Ratio. For example, in Figure 2.3f and 2.3g, we observe a decrease in accuracy when feature number increases using Trace Ratio.

2.6.2 Comparison with other LDA methods

We compare kaLDA with six other different methods, including LDA, Trace Ratio (TR), spdLDA, Maximum Margin Criteria (MMC), regularized LDA (RLDA), and Orthogonal Centroid Method (OCM). All LDA will reduce data to $K-1$ dimension. KNN ($knn = 3$) will be applied to do the classification after data is projected into the selected subspace. The other algorithms have already been introduced in related work section. The final classification accuracy is the average of 5-fold cross validation, and is reported in Table 2.2. The first column "kaLDA" reports kaLDA classification accuracy. kaLDA has the highest accuracy on 4 out of 8 datasets, including Caltec20, MSRC-MOM, ATT and Mnist. For Umist and Pie, kaLDA results

are very close to the highest accuracy. Overall, kaLDA performs better than all other methods.

2.6.3 Multi-label Classification

Six multi-label datasets are used in this part. These datasets include images features, music emotion and so on. Table 2.3 summarizes the attributes of those datasets.

MSRC-MOM and **MSRC-SIFT** data set is provided by Microsoft Research in Cambridge. It includes 591 images of 23 classes. **MSRC-MOM** is the Moment invariants (MOM) feature of images and each image has 384 dimensions. **MSRC-SIFT** is the SIFT feature and each image has 240 dimensions. About 80% of the images are annotated with at least one classes and about three classes per image on average.

Barcelona data set contains 139 images with 4 classes, i.e., "building", "flora", "people" and "sky". Each image has at least two labels.

Emotion [27] is a music emotion data, which comprises 593 songs with 6 emotions. The dimension of Emotion is 72.

Yeast [28] is a multi-label data set which contains functional classes of genes in the Yeast Saccharomyces cerevisiae.

Scene [29] contains images of still scenes with semantic indexing. It has 2407 images from 6 classes.

We use 5-fold cross validation to evaluate classification performance of different algorithms. K-Nearest Neighbour (KNN) classifier is used after the subspace projection. The algorithms we compared in this section includes Multi-label informed Latent Semantic Indexing (MLSI), Multi-label Dimensionality reduction via Dependence Maximization (MDDM), Multi-Label Least Square (MLLS), Multi-label Linear

Discriminant Analysis (MLDA). These algorithms have been introduced in related work section.

We compare the performance of kaLDA and other algorithms using macro accuracy (Table 2.4), macro-averaged F1-score (Table 2.5) and micro-averaged (Table 2.6) F1-score. Accuracy and F1 score are computed using standard binary classification definitions. In multi-label classification, macro average is a standard class-wise average, and it is related to number of samples in each class. However, micro average gives equal weight to all classes [8]. kaLDA achieves highest classification accuracy on 5 out of 6 datasets. On the remaining MSRC-SIFT dataset, kaLDA result is very close to the best method MLDA and beat all rest methods. kaLDA achieves highest macro and micro F1 score on 5 out of 6 datasets. Furthermore, kaLDA has the second highest macro and micro F1 score on dataset MSRC-SIFT. Overall, kaLDA outperforms other multi-label algorithms in terms of classification accuracy and macro and micro F1 score.

2.7 Conclusion

In this chapter, we propose a new kernel alignment induced LDA (kaLDA). The objective function of kaLDA is very similar to classical LDA objective. The Stifel-manifold gradient descent algorithm can solve kaLDA objective efficiently. We have also extended kaLDA to multi-label problems. Extensive experiments show the effectiveness of kaLDA in both single-label and multi-label problems.

CHAPTER 3

HARMONIC MEAN LINEAR DISCRIMINANT ANALYSIS

3.1 Introduction

It is difficult to find patterns from high dimensional data and analyze high dimensional data, but there are more and more high dimensional data generated every day in this big data era [30]. One simple example is that the camera quality of smarter phone becomes better and better nowadays, which means the image taken is larger and larger and some images may take several megabytes in size. In image classification, a small image of size 100×100 pixels will have a $10,000$ dimension pixel vector representation. In biology science, high-dimensional gene expression data is used to predict tumors and other diseases [31]. High-dimensional data not only costs a lot of storage, but also costs a lot of computing resources. More importantly, it also affects the performance of machine learning and data mining algorithms.

For many high dimensional data, there is an underlying low-dimensional structure which can capture the latent attributes of the high-dimensional data. Dimensionality reduction algorithms have been proposed to extract important information and features to help analyze high dimensional data. Dimensionality reduction is important in many applications of statistics, pattern recognition and machine learning. Many methods have been proposed for dimensionality reduction, such as principal component analysis (PCA) [32] and linear discriminant analysis (LDA) [33] [17].

LDA is a popular supervised dimensionality reduction algorithm. To be specific, let $X \in \mathbb{R}^{p \times n}$ be the data matrix, and $X = (\mathbf{x}_1, \cdots, \mathbf{x}_n)$, where p is data dimension, n is number of data points. Let $G \in \mathbb{R}^{p \times k}$ be the transformation ma-

trix to a k-dimensional subspace. The between-class scatter matrix S_b, within-class scatter matrix S_w and total scatter matrix S_t is defined as:

$$S_b = \sum_{k=1}^{K} n_k(\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T, \tag{3.1}$$

$$W_k = \frac{1}{n_k} \sum_{\mathbf{x}_i \in k} (\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)^T, \tag{3.2}$$

$$S_w = \sum_{k=1}^{K} n_k W_k, \tag{3.3}$$

$$S_t = S_b + S_w, \tag{3.4}$$

where K is total class number, n_k is number of points in class k, $\mathbf{m}_k$ is the mean of class k, $\mathbf{m}$ is the mean of entire data set:

$$\mathbf{m}_k = \frac{\sum_{\mathbf{x}_i \in k} \mathbf{x}_i}{n_k}, \quad \mathbf{m} = \frac{\sum_{i=1}^{n} \mathbf{x}_i}{n}. \tag{3.5}$$

S_b, S_w and S_t are semi-positive definite matrices. Classical LDA finds a transformation matrix G by solving the problem:

$$\max_G \mathrm{Tr} \frac{G^T S_b G}{G^T S_w G}. \tag{3.6}$$

There are many other formulations of LDA. The essence of LDA is to maximize the between-class distance while minimizing within-class distance. To maximize between-class distance distance in the subspace of G, the following problem can be maximized:

$$\max_G \sum_{k=1}^{K} \|G^T(\mathbf{m}_k - \mathbf{m})\|^2 = \mathrm{Tr}(G^T S_b G). \tag{3.7}$$

To minimize the sum of within-class distance in the subspace of G, the following problem can be minimized:

$$\min_G \sum_{k=1}^{K} \sum_{\mathbf{x}_i \in k} \|G^T(\mathbf{x}_i - \mathbf{m}_k)\|^2 = \mathrm{Tr}(G^T S_w G). \tag{3.8}$$

To combine the two tasks together, this leads to another similar LDA objective function, Trace Ratio [11, 13, 34]:

$$\max_{G} \frac{\mathrm{Tr}(G^T S_b G)}{\mathrm{Tr}(G^T S_w G)}, \text{ s.t. } G^T G = I, \tag{3.9}$$

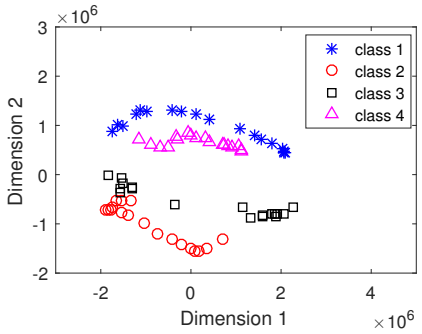
where constraint $G^T G = I$ ensures the columns of solution G are linearly independent. Null space based LDA (NLDA) [35] is another reformulation of LDA. Since classical LDA is not well defined when $G^T S_w G = \mathbf{0}$, in NLDA, the between-class distance is maximized in the null space of within-class scatter matrix S_w,

$$\max_{G} \mathrm{Tr}(G^T S_b G), \text{ s.t. } G^T S_w G = \mathbf{0}, G^T G = I \tag{3.10}$$

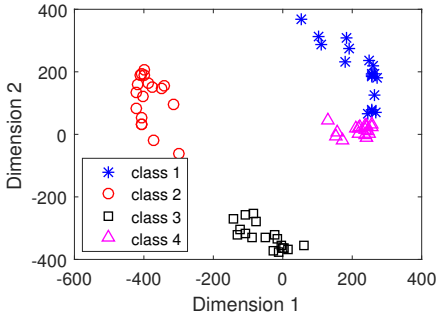
which is based on the idea that the null space of S_w contains sufficient discriminant information.

However, classical LDA, Trace Ratio and many reformulations of LDA have some limitations. First, they use arithmetic mean of between-class distances, which gives equal weights to all between-class distances, and large between-class distance could dominate the result. Second, they do not consider pairwise between-class distance and thus some classes may overlap with each other in subspace.

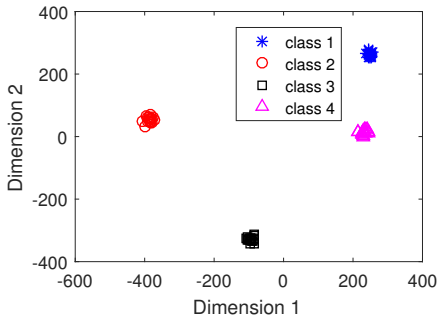
In this chapter, we propose two formulations of harmonic mean based Linear Discriminant Analysis: Harmonic Linear Discriminant HLDA (HLDA) and Harmonic Linear Discriminant Analysis pairwise (HLDAp), to demonstrate the benefit of harmonic mean between-class distance and overcome the limitations of classical LDA. The proposed HLDA and HLDAp differs in the way how the within-class distance is considered in the objective and HLDAp considers *pairwise* within-class distance. We also extend HLDA and HLDAp to multi-label classification problems. Finally, we present extensive experiments on single-label and multi-label data sets and investigate the performance of Harmonic Linear Discriminant Analysis with respect to subspace dimensions.



(a) PCA to 2D space.



(b) LDA to 2D space.



(c) Proposed Harmonic Linear Discriminant Analysis (HLDA) to 2D space.

Figure 3.1: Limitations of classical LDA.

The chapter is organized as follows: Section 3.2 discusses the limitations of classical LDA; Section 3.3 and 3.4 introduce the proposed HLDA and HLDAp objectives and algorithms to solve them; Section 3.5 uses several real data sets to demonstrate the effectiveness of HLDA and HLDAp, and compares them with PCA, classical LDA in 2-D space; Section 3.6 discusses the challenges in multi-label classification and proposes the multi-label version of HLDA and HLDAp; Section 3.7 presents the experiment results; Section 3.8 introduces the related work and the algorithms we compared in the experiment part; Section 3.9 concludes the chapter.

3.2 Limitations of Classical LDA

In Eq.(3.1), between-class distance of classical LDA is computed using the sum of distances between each class mean and total mean, in other words, using *arithmetic mean* of all between-class distances multiplying number of classes. However, this between-class distance has limitations. First, arithmetic mean of between-class distance gives equal consideration to all between-class distances, which makes larger between-class distances could dominate the objective function and thus limits the performance of LDA. Secondly, it does not consider pairwise between-class distance, and using distances between each class mean and *total mean* does not guarantee all pairwise classes are separated. In fact, using the *arithmetic mean* of pairwise between-class distances is equivalent to using distances between each class mean and *total mean*, which we will show in Lemma 3.

We use a small data set to show the two limitations of classical LDA. We take 4 classes from UMIST [36] data where each class has 18 points, and project them to 2D space. Figure 3.1a shows the result of using unsupervised PCA. Figure 3.1b shows the result in a 2D LDA space (using Eq.(3.6)). Points of different classes are more separated in LDA than in PCA. However, even though the sum of squared between-class distance is maximized, class 1 and class 4 are not well separated. First, large between-class distances dominates the arithmetic mean based LDA results. For example, distances between class 1 and 2, class 4 and 2 are large, distances between class 1 and 4 is very small. Second, LDA using Eq.(3.6) maximizes the distance between each class mean and total class mean, instead of pairwise between-class distance. Thus, class 1 and class 4 overlap with each other and pairwise between-class distance is not guaranteed to be maximized.

The limitations of arithmetic mean based between-class distance also exist in many other formulations of LDA, such as the null space LDA (NLDA), as in Eq.(3.10).

Similar to classical LDA, NLDA gives equal consideration to all between-class distances, which makes larger between-class distances could dominate the objective function and thus limits the performance of NLDA.

3.3 Harmonic Linear Discriminant LDA (HLDA)

In this section, we propose Harmonic Linear Discriminant Linear Discriminant Analysis (HLDA) objective using the harmonic mean based pairwise between-class distance to overcome limitations of classical LDA. We first use a lemma to show that Eq.(3.9) is equivalent to Eq.(3.17). Then we propose HLDA objective and an algorithm to solve the objective.

3.3.1 Objective Function

As we can see from the demonstration in Figure 3.1, pairwise between-class distance plays an important role in the projection. Figure 3.1c is a better solution than Figure 3.1b, because all classes in the solution are clearly separated and no two classes are too close to each other. In order to achieve this goal, we introduce the use of pair-wise between-class distance. To incorporate pairwise between-class distance into our objective, we define pairwise between-class scatter matrix $B_{k\ell}$ for class k and ℓ as:

$$B_{k\ell} = (\mathbf{m}_k - \mathbf{m}_\ell)(\mathbf{m}_k - \mathbf{m}_\ell)^T. \tag{3.11}$$

For ease of notations, let us define the following simplified sum notation:

$$\sum_{k<\ell} = \sum_{k=1}^{K-1} \sum_{\ell=k+1}^{K}, \quad \sum_{k,\ell} = \sum_{k=1}^{K} \sum_{\ell=1}^{K} \tag{3.12}$$

We now present Lemma 3 to show that Eq.(3.9) is equivalent to Eq.(3.17).

Lemma 3. *Using the definition of S_b in Eq.(3.1) and the definition of $B_{k\ell}$ in Eq.(3.11), we have the following identity:*

$$Tr(G^T S_b G) = \frac{1}{n}\sum_{k<\ell} n_k n_\ell \, Tr(G^T B_{k\ell} G), \tag{3.13}$$

where $n = \sum_k n_k$, n_k is number of samples in class k.

Proof. When $k = \ell$, $B_{k\ell} = \mathbf{0}$, so we can include $k = \ell$ in our following proof:

$$\sum_{k<\ell} n_k n_\ell \mathrm{Tr}(G^T B_{k\ell} G) = \frac{1}{2}\sum_{k,\ell} n_k n_\ell \mathrm{Tr}(G^T B_{k\ell} G)$$

$$\begin{aligned}
\mathrm{Tr}(G^T B_{k\ell} G) &= \mathrm{Tr}(G^T(\mathbf{m}_k - \mathbf{m} + \mathbf{m} - \mathbf{m}_\ell)(\mathbf{m}_k - \mathbf{m} + \mathbf{m} - \mathbf{m}_\ell)^T G) \\
&= \mathrm{Tr}(G^T(\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T G) && (3.14) \\
&+\mathrm{Tr}(G^T(\mathbf{m}_\ell - \mathbf{m})(\mathbf{m}_\ell - \mathbf{m})^T G) && (3.15) \\
&-2\mathrm{Tr}(G^T(\mathbf{m}_k - \mathbf{m})(\mathbf{m}_\ell - \mathbf{m})^T G) && (3.16)
\end{aligned}$$

In the equation above, for Eq.(3.14),

$$\begin{aligned}
&\sum_{k,\ell} n_k n_\ell \mathrm{Tr}(G^T(\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T G) \\
&= (\sum_\ell n_\ell)\mathrm{Tr}(G^T(\sum_k n_k(\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T)G) = n\mathrm{Tr}(G^T S_b G).
\end{aligned}$$

Eq.(3.15) is the same as Eq.(3.14).

For the Eq.(3.16), because $\sum_k n_k(\mathbf{m}_k - \mathbf{m}) = \mathbf{0}$,

$$\begin{aligned}
&\sum_{k,\ell} n_k n_\ell \mathrm{Tr}(G^T(\mathbf{m}_k - \mathbf{m})(\mathbf{m}_\ell - \mathbf{m})^T G) \\
&= \mathrm{Tr}[G^T(\sum_k n_k(\mathbf{m}_k - \mathbf{m}))(\sum_\ell n_\ell(\mathbf{m}_\ell - \mathbf{m})^T)G] = 0.
\end{aligned}$$

This completes the proof. □

From Lemma 3, Eq.(3.9) is identical to:

$$\max_G \sum_{k<\ell} \frac{n_k n_\ell}{n} \frac{\mathrm{Tr}(G^T B_{k\ell} G)}{\mathrm{Tr}(G^T S_w G)}, \text{ s.t. } G^T G = I. \tag{3.17}$$

Let $X_{k\ell} = \mathrm{Tr}(G^T B_{k\ell} G)/\mathrm{Tr}(G^T S_w G)$. The weighted arithmetic mean of $X_{k\ell}$ is

$$\langle X \rangle_{\text{arith}} = \frac{\sum_{k<\ell} n_k n_\ell X_{k\ell}}{\sum_{k<\ell} n_k n_\ell}, \tag{3.18}$$

Since $\sum_{k<\ell} n_k n_\ell$ = constant, Eq.(3.17) is the maximization of the arithmetic mean of $X_{k\ell}$'s.

It is clear that the arithmetic mean is dominated by large $X_{k\ell}$'s. Large $X_{k\ell}$ means class k is well-separated from class ℓ. However, it is the small $X_{k\ell}$'s that we should focus on since small $X_{k\ell}$ means class k is very close to class ℓ.

The weighted harmonic mean is given as

$$\langle X \rangle_{\text{harm}} = 1 \Big/ \left[\frac{\sum_{k<\ell} n_k n_\ell / X_{k\ell}}{\sum_{k<\ell} n_k n_\ell} \right]. \tag{3.19}$$

It is clear that small $X_{k\ell}$'s dominate the harmonic mean. In other words, *harmonic mean* focuses on (emphasize) the correct or critical parts that we wish to maximize.

For this reason, we propose to maximize the *harmonic mean* of pairwise between-class distances. Maximizing the harmonic mean Eq.(3.19) is equivalent to minimizing

$$\sum_{k<\ell} n_k n_\ell / X_{k\ell}. \tag{3.20}$$

This leads to our desired objective function of Harmonic Linear Discriminant Analysis (HLDA):

$$\min_G J_1(G) = \sum_{k<\ell} n_k n_\ell \frac{\mathrm{Tr}(G^T S_w G)}{\mathrm{Tr}(G^T B_{k\ell} G)}, \text{ s.t. } G^T G = I. \tag{3.21}$$

In summary, HLDA is proposed to weight more heavily the close distance pairs of classes in the optimization, the difficult part of the discriminant function; whereas

the standard LDA weights more of large distance pairs, the less important part of the discriminant function. Thus HLDA formulation is more robust.

Figure 3.1c shows the result of HLDA to project UMIST data to 2D space. Compared to PCA (Figure 3.1a) and LDA (Figure 3.1b), all classes are separated clearly and the within-class distance is minimized simultaneously.

3.3.2 Algorithm

We introduce an efficient algorithm to minimize HLDA objective. The gradient of Eq.(3.21) is given as:

$$\nabla J_1 \triangleq \frac{\partial J_1}{\partial G} = 2\sum_{k<\ell} n_k n_\ell \frac{S_w G}{\mathrm{Tr}(G^T B_{k\ell} G)} - 2\sum_{k<\ell} n_k n_\ell B_{k\ell} G \frac{\mathrm{Tr}(G^T S_w G)}{(\mathrm{Tr} G^T B_{k\ell} G)^2}. \tag{3.22}$$

Constraint $G^T G = I$ enforces G on the Stiefel manifold. Variations of G on this manifold is parallel transport, which gives some restriction to the gradient. This has been been worked out in [12]. The gradient that reserves the manifold structure is

$$\nabla J_1 - G[\nabla J_1]^T G. \tag{3.23}$$

Thus the algorithm computes the new G is given as follows:

$$G \leftarrow G - \eta(\nabla J_1 - G[\nabla J_1]^T G), \tag{3.24}$$

where η is step size. Due to fact that the manifold preserving gradient of Eq.(23) only enforces the condition $G^T G = I$ to first order, after every 10-20 iterations, we bring G back to the manifold using SVD decomposition. Mathematically, let $\mathrm{SVD}(G) = U\Sigma V^T$. Then the manifold preserving $G = UV^T$. Since size of G is $p \times k$ and k is subspace dimension which is typically small, this SVD step is very fast. Algorithm 2 summarizes the steps to solve Eq.(3.21). The objective is optimized in an iterative fashion. There is no need to do Eigen decomposition or matrix inverse for scatter matrices.

Algorithm 2 Stiefel gradient descent algorithm for HLDA.

Input: Data matrix $X \in \mathbb{R}^{p\times n}$ with n data points in p dimensional space; class indicator matrix $Y \in \mathbb{R}^{n\times K}$, K is number of classes; subspace dimension k

Output: Projection matrix $G \in \mathbb{R}^{p\times k}$

1: Initialize G

2: Compute S_w and $B_{k\ell}$ using Eq.(3.3) and Eq.(3.11)

3: **while** Objective value Eq.(3.21) not converge **do**

4: Compute Stiefel manifold gradient using Eq.(3.23)

5: Update G using Eq.(3.24)

6: **end while**

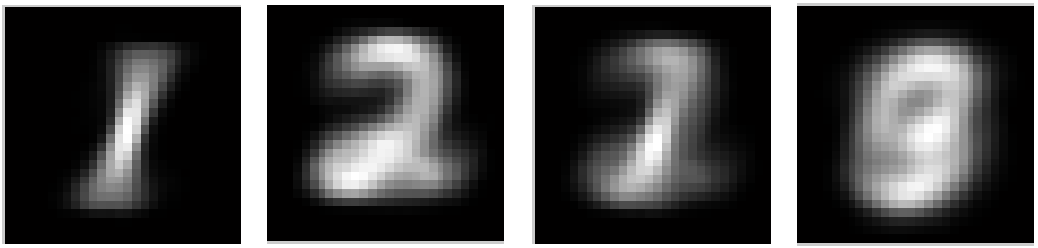
While initializing matrix G, if subspace dimension $k <= K-1$, we can use classical LDA Eq.(3.6) solution to initialize G; when $k > K-1$, we can use trace ratio LDA Eq.(3.9) solution to initialize G. This ensures that our approach can find a better solution than other LDA formulations (see experiments part for comparison).

3.3.3 Comparison to SUM version HLDA

In HLDA, it is also possible to move the within-class distance part $\mathrm{Tr}(G^T S_w G)$ from the nominator to a separate term as the following

$$\begin{aligned} &\min_G \ \gamma \mathrm{Tr}(G^T S_w G) + \sum_{k<\ell} \frac{n_k n_\ell}{\mathrm{Tr}(G^T B_{k\ell} G)}, \\ &\text{s.t. } G^T G = I, \end{aligned} \tag{3.25}$$

The advantage here is that the relative weight of the two tasks can be explicitly controlled by the parameter γ, while in HLDA the relative weight of the two tasks are prefixed. When $\gamma \to \infty$, Eq.(3.25) focuses on minimizing within-class distance only, which is equal to finding the null space of within-class scatter matrix S_w. When $\gamma \to 0$,



(a) “1”. (b) “2”. (c) “1” and “2”. (d) “0” to “9”.

Figure 3.2: Mean of digit images.

Eq.(3.25) focuses on maximizing pairwise between-class distance. However, the tuning of extra parameter γ add significant computational time in real applications. This framework has been studied in [37], and will not be discussed further in this chapter.

3.4 Harmonic Linear Discriminant Analysis pairwise (HLDAp)

In many datasets, different class has different within-class covariance, the global average of within-class S_w used in Eq.(3.21) would differ significantly from each class. However, in general, the average of two classes are likely to be close to each of the two classes.

For example, consider handwritten digits data MNIST. We take 500 images from each class. Figure 3.2 shows the mean of digit “1”, mean of digit “2”, mean of digits “1” and “2”, and mean of digits “0” to “9”. We can see that the mean of digits “1” and “2” retains some similarity with digits “1” and “2”, whereas the mean of digits “0” to “9” is very different from digits “1” and “2”.

Using this idea, we believe that using the global average of within-class distances (variances) of all classes is a less accurate representation as compared to use average of two class covariances. Fortunately, this pairwise average can be accommodated into

the framework of Eq.(3.21). For this purpose, we introduce the pair-wise within-class covariance (scatter matrix) of class k and ℓ

$$W_{k\ell} = \frac{1}{n_k + n_\ell}(n_k W_k + n_\ell W_\ell), \tag{3.26}$$

where W_k and W_ℓ can be given from Eq.(3.2).

The objective function of HLDA is then changed to:

$$\begin{aligned} &\min_G J_2(G) = \sum_{k<\ell} n_k n_\ell \frac{\mathrm{Tr}(G^T W_{k\ell} G)}{\mathrm{Tr}(G^T B_{k\ell} G)}, \\ &\text{s.t. } G^T G = I, \end{aligned} \tag{3.27}$$

where constraint $G^T G = I$ ensures the columns of solution G are linearly independent. We call Eq.(3.27) Harmonic Linear Discriminant Analysis pairwise (HLDAp).

Again, we use Stiefel gradient descend method to solve the minimization problem. The gradient of Eq.(3.27) is:

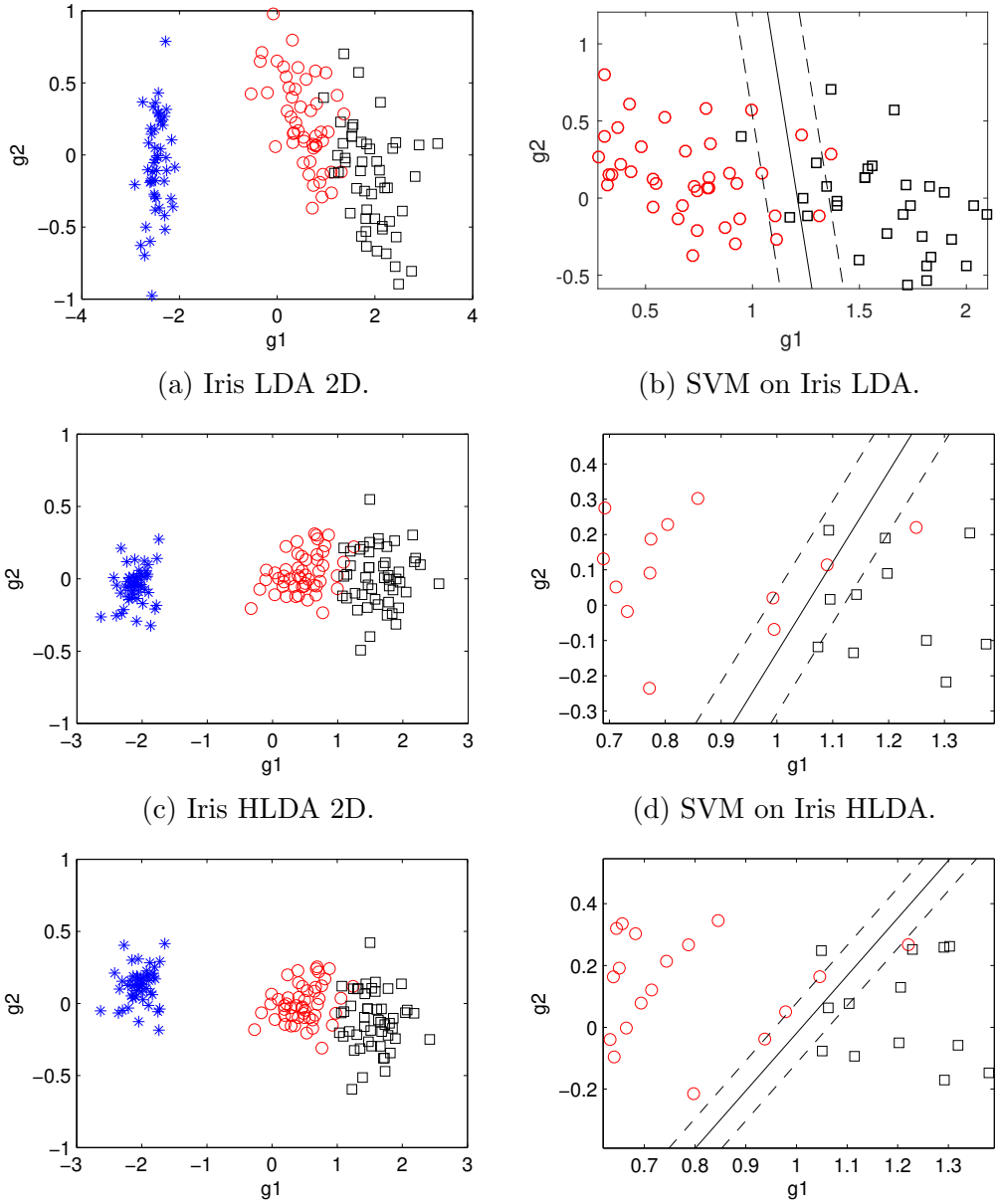
$$\nabla J_2 \triangleq \frac{\partial J_2}{\partial G} = \sum_{k<\ell} 2 n_k n_\ell [\frac{W_{k\ell} G}{\mathrm{Tr}(G^T B_{k\ell} G)} - B_{k\ell} G \frac{\mathrm{Tr}(G^T W_{k\ell} G)}{(\mathrm{Tr} G^T B_{k\ell} G)^2}]. \tag{3.28}$$

We then use the natural gradient of Eqs.3.23, 3.24) to enforce G on the Stiefel manifold.

3.5 Illustration

To show the effectiveness of HLDA and HLDAp, Figure 3.3, 3.4, 3.5 and 3.6 visualize real data sets, Iris [38], PIE, YaleB and ATT, in 2-D subspace.

Iris data has 150 samples in total, 3 classes and dimension $p = 4$. Figure 3.3a, 3.3c, 3.3e show the classical LDA, HLDA and HLDAp 2D projection of Iris data respectively. The red circle and black square class are very close in these figures. Figure 3.3b, 3.3d, 3.3f show the SVM results on two classes (red circle and black



(a) Iris LDA 2D.

(b) SVM on Iris LDA.

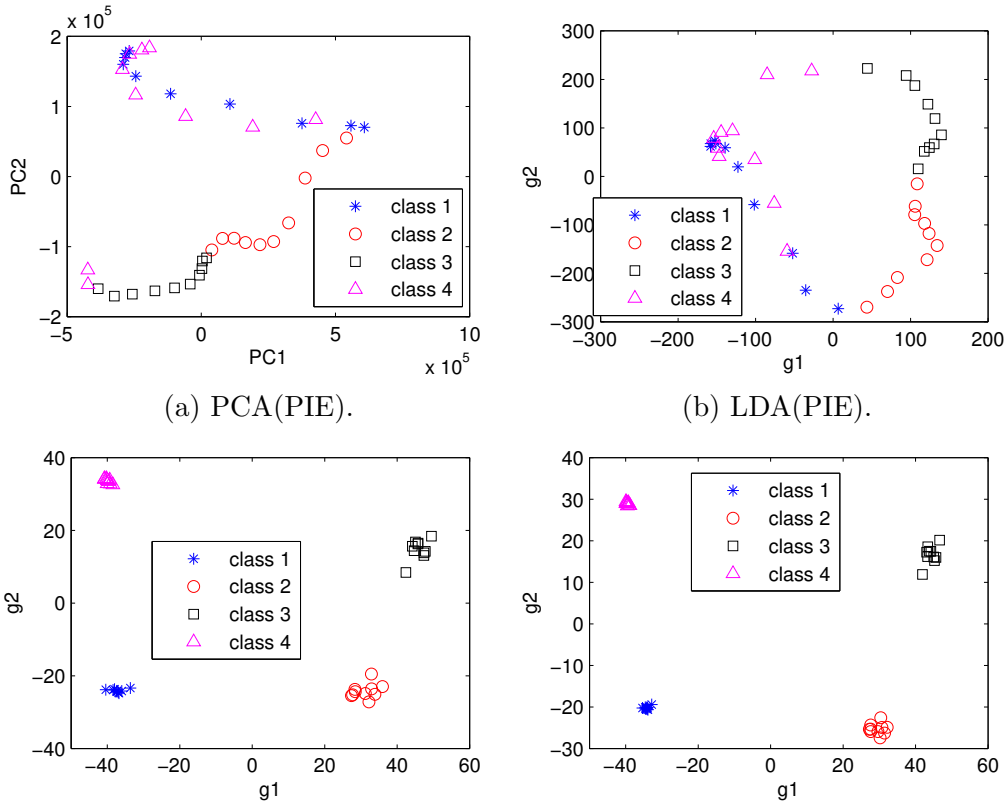
(c) Iris HLDA 2D.

(d) SVM on Iris HLDA.

(e) Iris HLDAp 2D.

(f) SVM on Iris HLDAp.

Figure 3.3: Illustration of Iris data (dimension $p = 4$, sample number $n = 150$ and class number $K = 3$) in 2-D subspace using LDA, HLDA and HLDAp, $g1$ and $g2$ are the two subspace dimensions. Figure 3.3b, 3.3d and 3.3f show SVM results on red circle class and black square class: Figure 3.3b has 5 misclassified samples; Figure 3.3d and 3.3f have 2 misclassified samples respectively.



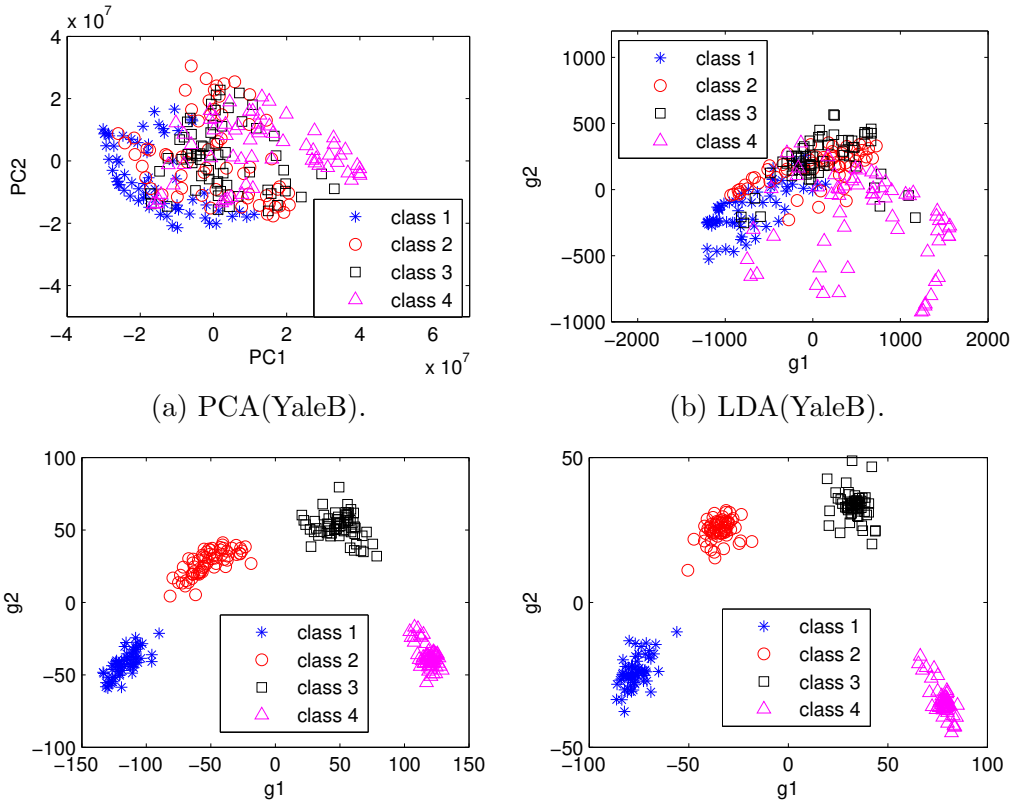
(a) PCA(PIE). (b) LDA(PIE).

(c) HLDA(PIE). (d) HLDAp(PIE).

Figure 3.4: Visualization of PIE demo data (dimension $p = 1024$, sample number $n = 40$ and class number $K = 4$) in 2-D subspace, $g1$ and $g2$ are the two subspace dimensions, $PC1$ and $PC2$ are the two principle components of PCA.

square) of Iris data. Figure 3.3b has 5 misclassified samples. Figure 3.3d and 3.3f have 2 misclassified samples respectively.

Figure 3.4, 3.5 and 3.6 show 2-D projection of PCA, LDA, HLDA and HLDAp on demo data PIE, YaleB and ATT (see Table 3.1 for more information about these data). In this demo, we take 4 classes from each data. From Figures (3.4a, 3.4b, 3.5a, 3.5b, 3.6a, 3.6b), we can see that data points from 4 classes are mixed together and it is difficult to separate any two classes from the the figures. Figures (3.4c, 3.4d, 3.5c, 3.5d, 3.6c, 3.6d) show the project results using Eq.(3.21) and Eq.(3.27). For all three demo data, 4 classes are clearly separated and there are no overlaps.



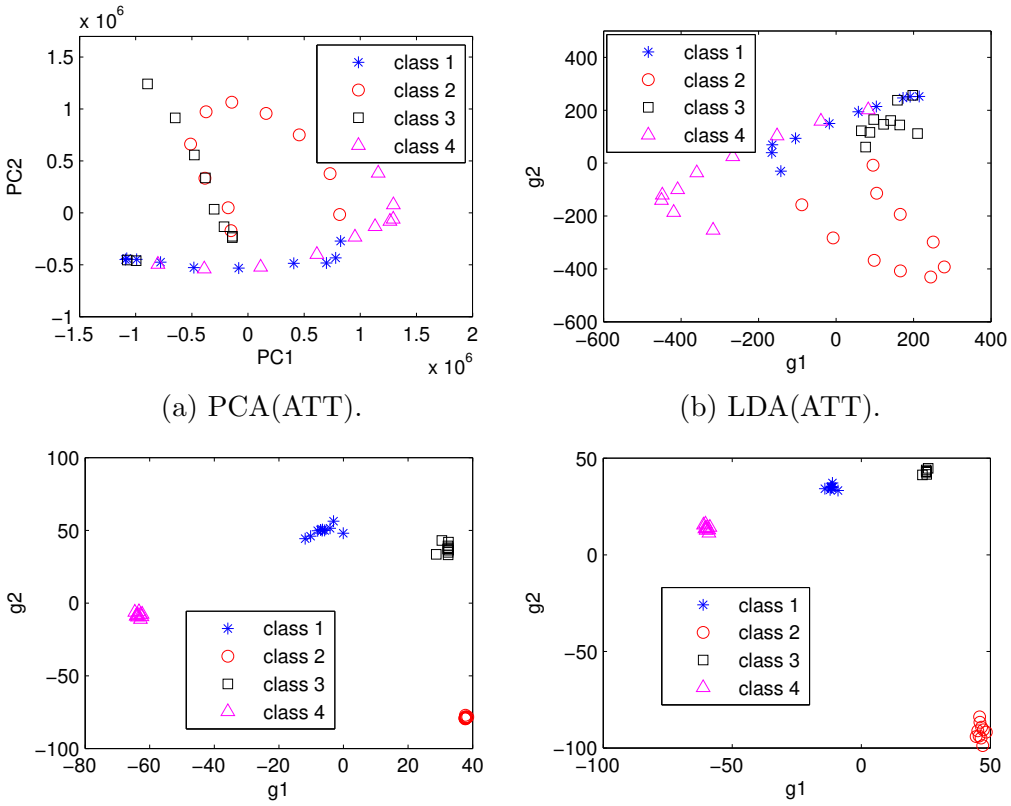
(a) PCA(YaleB). (b) LDA(YaleB).

(c) HLDA(YaleB). (d) HLDAp(YaleB).

Figure 3.5: Visualization of YaleB demo data (dimension $p = 504$, sample number $n = 256$ and class number $K = 4$) in 2-D subspace, $g1$ and $g2$ are the two subspace dimensions, $PC1$ and $PC2$ are the two principle components of PCA.

3.6 Multi-label HLDA and HLDAp

In image and video annotation, each image is usually associated with several different conceptual classes. Let's take two sample images from MSRC data in Figure 3.7 as an example. Figure 3.7a is annotated using 3 words: sky, plane and grass; Figure 3.7b is annotated using 3 words: car, building, road. In machine learning, such problem that requires each data point to be assigned to multiple different classes is called multi-label classification problem. In contrast, in traditional single-label classification, which is also called single-label multi-class classification, each data



(a) PCA(ATT). (b) LDA(ATT).

(c) HLDA(ATT). (d) HLDAp(ATT).

Figure 3.6: Visualization of ATT demo data (dimension $p = 644$, sample number $n = 40$ and class number $K = 4$) in 2-D subspace, $g1$ and $g2$ are the two subspace dimensions, $PC1$ and $PC2$ are the two principle components of PCA.

point is only classified into one category. Multi-label multi-class problem is more generalized than single-label multi-class problem.

An important difference between single-label classification and multi-label classification is that class memberships in single-label classification are mutually exclusive, while class memberships in multi-label classification are overlapped with 2 or more classes. Class memberships can be inferred from label correlations, which can be used to improve classification. It has stimulated many multi-label learning algorithms [20] [21] [22] [8].

(a) sky, plane, grass. (b) car, building, road.

Figure 3.7: Sample images from MSRC data set. Each image is annotated with several different words. In a multi-label multi-class classification problem, each image is classified into more than 1 class.

However, Linear Discriminant Analysis (LDA) by nature is derived for single-label classification. Wang proposed a multi-label formulation of scatter matrices for multi-label data in [8]. Multi-label class indicator matrix $Y \in \mathbb{R}^{n \times K}$ is defined as

$$Y_{ik} = \begin{cases} 1, & \text{if point } i \text{ is in class } k. \\ 0, & \text{otherwise.} \end{cases} \tag{3.29}$$

For data point i, $\sum_k Y_{ik} > 1$, which means that data i belongs to more than 1 class. Multi-label between-class scatter matrix $\widetilde{S_b}$ and within-class scatter matrix $\widetilde{S_w}$ are defined as follows [8]:

$$\widetilde{S_b} = \sum_{k=1}^{K} \sum_{i=1}^{n} Y_{ik} (\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T, \tag{3.30}$$

$$\widetilde{S_w} = \sum_{k=1}^{K} \sum_{i=1}^{n} Y_{ik} (\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)^T, \tag{3.31}$$

where $\mathbf{m}_k$ is the mean of class k and $\mathbf{m}$ is global mean, defined as follows:

$$\mathbf{m}_k = \frac{\sum_{i=1}^{n} Y_{ik} \mathbf{x}_i}{\sum_{i=1}^{n} Y_{ik}}, \quad \mathbf{m} = \frac{\sum_{k=1}^{K} \sum_{i=1}^{n} Y_{ik} \mathbf{x}_i}{\sum_{k=1}^{K} \sum_{i=1}^{n} Y_{ik}}. \tag{3.32}$$

Eqs.(3.30,3.31) are also equivalent to Eq.(28, 29, 30) in [39].

Inspired from Eqs.(3.30,3.31), we can define multi-label pair-wise between-class scatter matrix $\widetilde{B}_{k\ell}$ for class k and ℓ as:

$$\widetilde{B}_{k\ell} = (\mathbf{m}_k - \mathbf{m}_\ell)(\mathbf{m}_k - \mathbf{m}_\ell)^T. \tag{3.33}$$

$$n_k = \sum_{i=1}^{n} Y_{ik},\ n_\ell = \sum_{i=1}^{n} Y_{i\ell}. \tag{3.34}$$

We also define the multi-label within-class scatter matrix $\widetilde{W}_k$ for class k as:

$$\widetilde{W}_k = \frac{1}{n_k} \sum_{i=1}^{n} Y_{ik}(\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)^T. \tag{3.35}$$

3.6.1 Multi-label HLDA

Using Eqs.(3.31, 3.32, 3.33, 3.34), the objective of Multi-label HLDA can be proposed as:

$$\begin{aligned} &\min_G \sum_{k<\ell} n_k n_\ell \frac{\mathrm{Tr}(G^T \widetilde{S_w} G)}{\mathrm{Tr}(G^T \widetilde{B}_{k\ell} G)}, \\ &\text{s.t.} \quad G^T G = I. \end{aligned} \tag{3.36}$$

Eq.(3.36) can be solved using similar approach as Eq.(3.21).

3.6.2 Multi-label HLDAp

Using Eqs.(3.35), let us define the multi-label pair-wise within-class scatter matrix $\widetilde{W}_{k\ell}$ as:

$$\widetilde{W}_{k\ell} = \frac{1}{n_k + n_\ell}(n_k \widetilde{W}_k + n_\ell \widetilde{W}_\ell). \tag{3.37}$$

Using Eqs.(3.33, 3.34, 3.37), the objective of Multi-label HLDAp can be proposed as:

$$\begin{aligned} &\min_G \sum_{k<\ell} n_k n_\ell \frac{\mathrm{Tr}(G^T \widetilde{W}_{k\ell} G)}{\mathrm{Tr}(G^T \widetilde{B}_{k\ell} G)}, \\ &\text{s.t.} \quad G^T G = I, \end{aligned} \tag{3.38}$$

Eq.(3.38) can be solved using similar approach as Eq.(3.27).

Table 3.1: Experiment single-label dataset.

Data	dimension p	sample number n	class number K
UMIST	644	360	20
PIE	1024	680	68
YaleB	504	1984	31
ATT	644	400	40
MNIST	784	1000	10
ISOLET2	617	1560	26
ISOLET3	617	1560	26

Table 3.2: Experiment multi-label dataset.

Data	dimension p	sample number n	class number K
MediaMill	120	6601	74
Barcelona	48	139	4

3.7 Experiments

In this section, we compare the performance of proposed harmonic mean between-class distance based HLDA and HLDAp with other LDA formulation algorithms and perform experiments on single-label and multi-label problems. We will show the convergence and efficiency of proposed algorithm. We will systematically study the relationship of classification performance with subspace dimension number k.

3.7.1 Data

We use 7 single-label datasets and 2 multi-label datasets in this experiment. These datasets come from different domains, such as face image, handwritten digits, speech recognition and multimedia videos. Data attributes are summarized in Table 3.1 and Table 3.2.

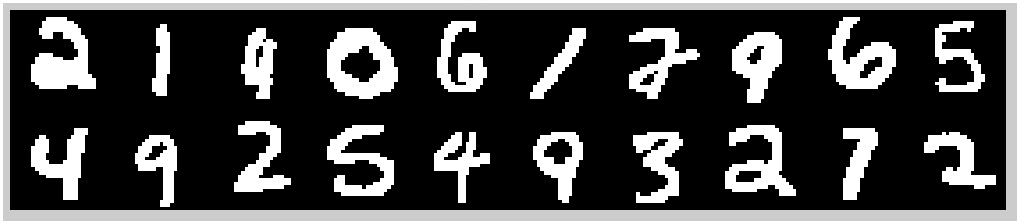
Single-label data UMIST [36] is a dataset of 360 face images (Sheffield Face database) taken from 20 persons with mixed race, gender and appearance. Each person has 18 images which were resized to 28x23 (644 pixels or dimensions). PIE [25] is a face database from Carnegie Mellon Robotics Institute. In total, it has

(a) UMIST

(b) PIE

(c) YaleB

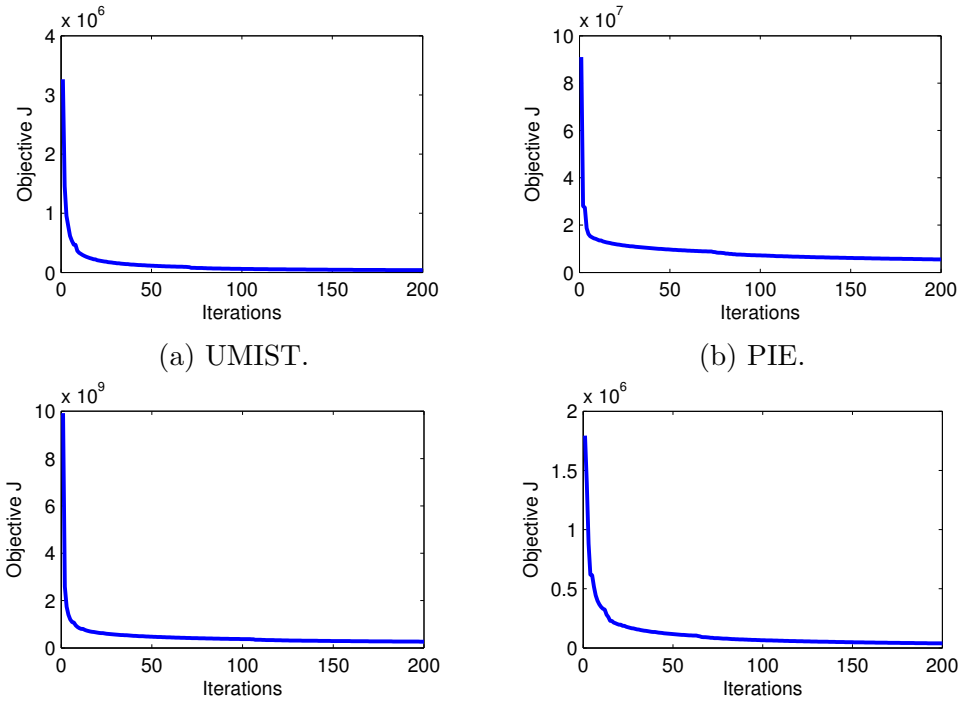
(d) ATT



(e) MNIST

Figure 3.8: Experiment example images.

68 different persons and 10 images for each person with different poses, different illumination conditions, and different expressions. Images were resized to 32x32 (1024 pixels). YaleB [40] contains images of 31 persons under 9 poses and 64 illumination conditions. Each person has 64 images with size 24x21 (504 pixels). ATT [41] data contains 400 images of 40 persons, with 10 images for each person. The images have been resized to 28x23 (644 pixels). MNIST [26] is a handwritten digits dataset with



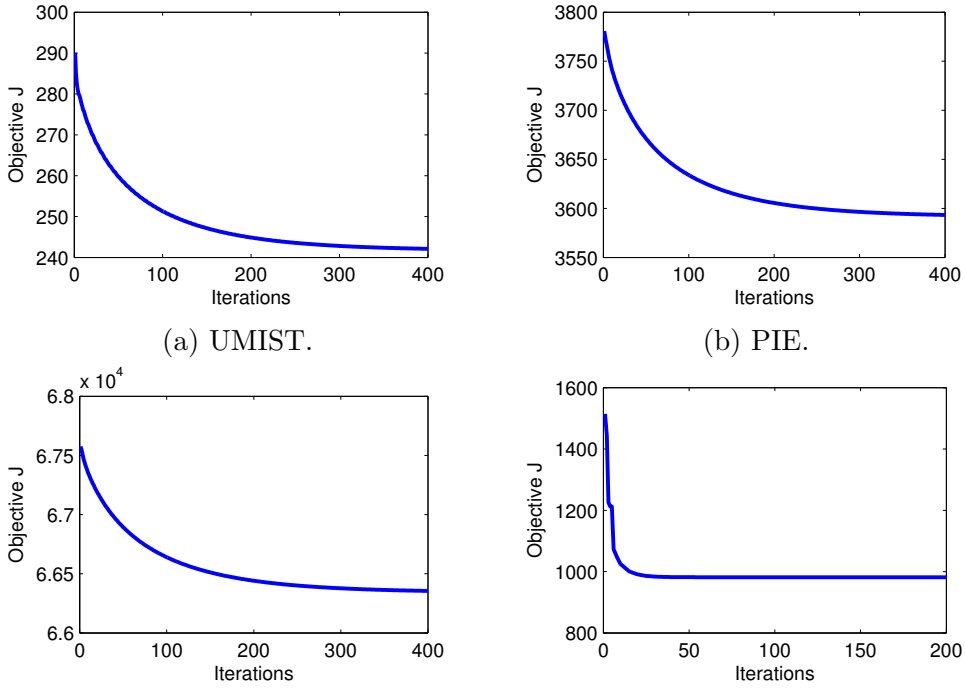
(a) UMIST. (b) PIE.

(c) YaleB. (d) ATT.

Figure 3.9: HLDA algorithm convergence (Algorithm 2, objective Eq.(3.21)).

10 classes and 100 images of size 28x28 (784 pixels) for each class. In ISOLET2 (Isolated Letter Speech Recognition) [38] data, each English letter was read twice by 30 people. Each recorded voice for a letter was analyzed and data (such as amplitudes, zero-crossing rates, DFT coefficients, etc.) are collected to form a feature vector of 617 dimensions. There are 26 classes. Each has 60 samples. ISOLET3 is the same as ISOLET2, but with different speakers. Figure 3.8 shows some example images of single label data.

Multi-label data MediaMill data [42] is a multi-label data from video concept detection problems. It has 74 classes and 6601 samples. Barcelona data [7] contains image moments of 139 images with 4 classes, such as buildings, flora, people and sky. Each image has at least two labels.



(a) UMIST. (b) PIE.

(c) YaleB. (d) ATT.

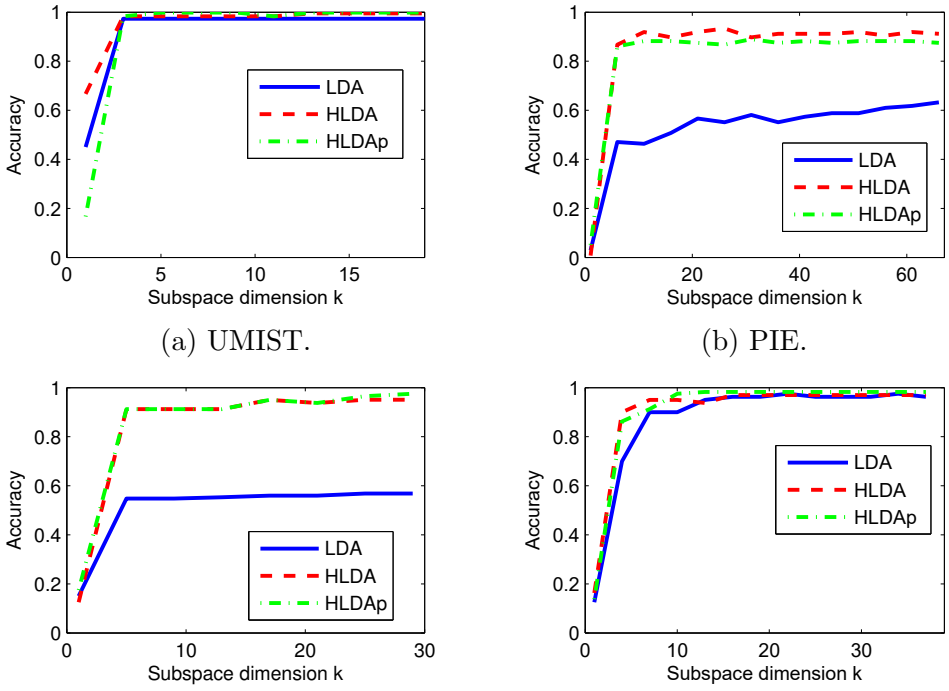
Figure 3.10: HLDAp algorithm convergence (objective Eq.(3.27)).

3.7.2 Convergence of Algorithm

We take 4 single-label datasets as examples to show the convergence speed of Algorithm 2. Figure 3.9 and Figure 3.10 show the objective Eq.(3.21) and Eq.(3.27) converge quickly in 200 to 400 iterations.

3.7.3 Effect of Subspace Dimension

We want to study the effect to subspace dimension k to the performance of HLDA and HLDAp. We take 4 datasets and apply on them HLDA, HLDAp and LDA with different subspace dimension k (from 1 to $K-1$). Then we use KNN as classifier to see the classification accuracy. Figure 3.11 shows the results. HLDA gives better accuracy than HLDAp and LDA on UMIST data when k is less than 3. HLDA and HLDAp is a little better than LDA when k is larger than 3. For data PIE, HLDA



(a) UMIST. (b) PIE.

(c) YaleB. (d) ATT.

Figure 3.11: Accuracy using different subspace dimension k (Check Table 3.3 for the improvement at $K-1$).

and HLDAp gives better accuracy than LDA. For YaleB, the improvement of HLDA and HLDAp over LDA is significant as well. For data ATT, HLDA and HLDAp gives better accuracy than LDA.

3.7.4 Single-Label Classification Experiment

The experiment use 5-fold cross validation to evaluate the classification performance of different algorithms when dimension is $k = K - 1$. K-Nearest Neighbour (KNN) classifier is then used after each dimension reduction algorithm. Table 3.3 shows the classification accuracy of different approaches, and HLDA and HLDAp give better results.

Table 3.3: Single-label experiment results (subspace dimension is $K-1$, best results are in bold).

Data	HLDA	HLDAp	LDA	TraceRatio	MMC	RLDA	ULDA	OLDA	OCM	OLSLDA	sdpLDA
UMIST	0.9950	**0.9983**	0.9733	0.9533	0.9817	0.9717	0.9733	0.9483	0.9717	0.9733	0.9717
PIE	**0.9007**	0.8805	0.6559	0.8456	0.8574	0.6265	0.6221	0.6485	0.6265	0.8250	0.6265
YaleB	0.9508	**0.9651**	0.5683	0.9112	0.9341	0.5009	0.5009	0.6630	0.5288	0.8435	0.5288
ATT	0.9700	**0.9825**	0.9675	0.8525	0.9675	0.9625	0.9625	0.9500	0.9625	0.9650	0.9625
MNIST	**0.8860**	0.8830	0.8770	0.7660	0.8650	0.8770	0.8720	0.8620	0.8770	0.7410	0.8770
ISOLET2	**0.9286**	0.9199	0.9154	0.5596	0.9199	0.8628	0.8737	0.8096	0.8929	0.8769	0.8929
ISOLET3	0.8917	**0.9151**	0.8859	0.5340	0.8981	0.8276	0.8385	0.7795	0.8641	0.8154	0.8641

Table 3.4: Multi-label experiment results (best results are in bold).

Data		HLDA	HLDAp	MLR	MLSI	MDDM	MLLS	MLDA
MediaMill	Accuracy	**0.9267**	0.9248	0.7705	0.8962	0.9044	0.8994	0.9036
	Macro F1	**0.5616**	0.5420	0.2546	0.5244	0.5593	0.5426	0.5571
	Micro F1	0.3988	0.3808	0.2239	0.4064	0.4432	0.4370	**0.4448**
Barcelona	Accuracy	**0.6962**	0.6779	0.6089	0.6436	0.6470	0.6524	0.6290
	Macro F1	**0.7683**	0.7534	0.6483	0.7286	0.7301	0.7341	0.7169
	Micro F1	**0.7257**	0.7081	0.5865	0.6891	0.6861	0.6904	0.6772

3.7.5 Multi-Label Classification Experiment

We compare the performance of Multi-label HLDA and Multi-label HLDAp with 5 other multi-label dimension reduction algorithms on 2 multi-label datasets in terms of macro accuracy, macro-averaged F1-score and micro-averaged F1-score. Macro-average is the average based on the overall testing dataset, while micro-average is the average which gives equal weight to each class. Macro-averaged and micro-averaged F1-score are widely used as a metric to evaluate classification performance [43]. MLSI, MDDM, MLLS, MLDA will be introduced in related work section (section 3.8). MLR is muli-label linear regression, which uses the closed-form solution of standard linear regression. This experiment uses 5-fold cross validation to evaluate the classification performance of different algorithms when dimension is $k = K-1$. K-Nearest Neighbour (KNN) classifier is then used after each algorithm. Table 3.4 shows that Multi-label HLDA and Multi-label HLDAp give better results over other algorithms.

3.8 Related Work

Researchers and engineers nowadays have larger and larger data with very high dimension to be processed everyday [30]. Many big data technologies including cloud computing, dimension reduction, accelerating algorithms have been proposed [44, 45, 46, 47, 7, 48]. Trace ratio problem has been studied thoroughly in recent years. Many dimension reduction algorithms can be reduced to a trace ratio objective. But trace ratio problem does not have closed-form solution. Thus how to solve trace ratio efficiently becomes an interesting research topic. Wang [11] proposed an efficient iterative algorithm to get an approximate solution. Shen [49] proposed a formulation for solving the trace ratio problem directly. Nie proposed a Trace Ratio criteria for feature selection[50]. Each feature subset has a feature score, which is computed by trace ratio. They propose an iterative algorithm to find the global optimal feature subset. A number of LDA reformulation ideas have be proposed in recent years, such as PCA+LDA [51], regularized LDA(RLDA) [14], null space LDA (NLDA) [35], Orthogonal Centroid Method (OCM) [15], Uncorrelated LDA(ULDA)[16], Orthogonal LDA (OLDA)[16], etc. Ye introduced a unified framework for generalized LDA in [17]. The unified framework consists of four steps:

1. Compute the eigenvalues $\{\lambda_i\}_{i=1}^d$ and eigenvectors $\{u_i\}_{i=1}^d$ of total covariance matrix S_t, where d is the dimension of data. So $S_t = \sum_{i=1}^d \lambda_i u_i u_i^T$.
2. Given a transfer function ϕ: $\tilde{\lambda}_i = \phi(\lambda_i)$. Construct $\tilde{S}_t = \sum_{i=1}^d \tilde{\lambda}_i u_i u_i^T$.
3. Compute the eigenvectors of matrix $\tilde{S}_t^+ S_b$ that correspond to the largest q eigenvalues, where q is the rank of S_b and $\tilde{S}_t^+$ means pseudo-inverse of $\tilde{S}_t$. Construct matrix G using these q eigenvectors.
4. Optional: compute the QR decomposition of $G = QR$.

The final projection is given as G or Q. In RLDA, the transfer function is $\phi(\lambda_i) = \lambda_i + \mu$. In ULDA, $\phi(\lambda_i) = \lambda_i$ and the optional QR decomposition is not

applied. In OLDA, $\phi(\lambda_i) = \lambda_i + \mu$ and the optional QR decomposition is applied. In OCM, the optimal transformation is the top eigenvectors of S_b and the transfer function is $\phi(\lambda_i) = 1$. Maximum Margin Criteria (MMC) [18] finds a projection G to maximize $\mathrm{Tr}(G^T(S_b - S_w)G)$. Semi-Definite Positive LDA (sdpLDA) [19] solves the maximization of $\mathrm{Tr}(G^T(S_b - \lambda_1 S_w)G)$, where λ_1 is the largest eigenvalue of $S_w^{-1}S_b$.

In the following, we introduce some related work about multi-label dimension reduction.

MLSI. Multi-label informed Latent Semantic Indexing (MLSI) [20] makes use of supervision information to solve the problem of

$$\begin{aligned} &\max_G \mathrm{Tr}(G^T((1-\beta)XX^TXX^T + \beta XYY^TX^T)G), \\ &\text{s.t. } G^TXX^TG^T = I, \end{aligned}$$

where the first term is the original Latent Semantic Indexing objective and the second term is the supervised term.

MDDM. Multi-label Dimensionality reduction via Dependence Maximization (MDDM) [21] finds a subspace by solving the following problem:

$$\max_G \mathrm{Tr}(G^TXHYY^THX^TG),$$

where $H = I - \mathbf{e}\mathbf{e}^T/n$ is the centralizing matrix. MDDM maximizes the dependence between the original features and associated class labels.

MLLS. Multi-Label Least Square (MLLS) [22] tries to find a subspace through solving the following problem:

$$\max_G \mathrm{Tr}(G^T(I - \alpha M)^{-1}(M_{-1}XYY^TX^TM_{-1})G),$$

$$M = \frac{1}{n}XX^T + (\alpha + \beta)I.$$

MLDA. Multi-label Linear Discriminant Analysis (MLDA) [8] finds a projection matrix G to maximize an objective function which is very similar to classical single label LDA:

$$\max_G \mathrm{Tr} \frac{G^T \widetilde{S}_b G}{G^T \widetilde{S_w} G},$$

where $\widetilde{S}_b$ and $\widetilde{S_w}$ can be computed from Eq.(3.30) and Eq.(3.31).

3.9 Conclusion

In this chapter, we proposed two formulations of harmonic mean based Linear Discriminant Analysis: Harmonic Linear Discriminant Analysis (HLDA) and Harmonic Linear Discriminant Analysis pairwise (HLDAp), to overcome the limitations of classical LDA. HLDA and HLDAp make use of weighted harmonic mean of pairwise between-class distance and gives higher priority to maximize small between-class distances. We extended HLDA and HLDAp to multi-label classification problems. Extensive experiments of HLDA and HLDAp on single-label and multi-label data sets show that HLDA and HLDAp have better performance than approaches using arithmetic mean based between-class distance and the potential benefit of harmonic mean based LDAs.

CHAPTER 4

MULTI-VIEW LOW-RANK REGRESSION

4.1 Introduction

In many tasks, a single object can be described using information from different channels (or views). For example, a 3-D object can be described using pictures from different angles; a website can be described using the words it contains, and the hyperlinks it contains; an image can be described using different features, such as SIFT feature, and HOG feature; in daily life, a person can be characterized using age, height, weight and so on. These data all comes from different aspects and channels. Multi-view problems aim to improve existing single view model by learning a model utilizing data collected from multiple channels [52] [53] [54].

Low-rank regression model has been proved to be an effective learning mechanism by exploring the low-rank structure of real life data [55] [56] [57]. Existing regression models only work on single view data. To be specific, linear regression finds a linear model with respect to the single view feature data to fit target class data [58]. Let matrix $B \in \mathbb{R}^{p\times c}$ be the parameter of the linear model. Linear regression solves a problem of $\min_B ||Y - X^T B||_F^2$, where $X = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n] \in \mathbb{R}^{p\times n}$ is the single view feature data matrix and $Y \in \mathbb{R}^{n\times c}$ is the target class indicator matrix. Ridge regression can achieve better results by adding a Frobenius norm based regularization on linear regression loss objective [59] [60]. Ridge regression solves the problem $\min_B ||Y - X^T B||_F^2 + \lambda||B||_F^2$, where λ is the regularization weight parameter. Cai [57] showed that when B is low-rank, regression is equivalent to linear discrim-

inant analysis based regressions. However, all these work only works for single-view problems.

In this chapter, we propose a multi-view low-rank regression model [48] by imposing low-rank constraints on regression model. This model can be solved using closed-form solution directly. In linear regression, low rank parameter matrix B^ν is dependent on view ν. Through theoretical analysis, we show that multi-view low-rank regression model is equivalent to do regression in the subspace of each view. In other words, let $B^\nu = A_\nu B$, and it is equivalent to find the shared regression parameter matrix B under the subspace transformation A_ν with respect to view ν. Extensive experiments performed on 4 multi-view datasets show that the proposed model outperforms single-view regression model and reveals that low-rank structure can improve the classification result of a full-rank model.

Notations. In this chapter, matrices are written in uppercase letters, such as X, Y. Vectors are written in bold lower case letters, such as $\mathbf{x}$, $\mathbf{y}$. $\mathrm{Tr}(X)$ means the trace operation for matrix X.

4.2 Multi-view Low Rank Regression

Assume that there are v views and c classes, p_ν is the dimension of view ν, n_j is the sample size of the j-th class, and n is the total sample size. Let $X_\nu = [\mathbf{x}_1^\nu, ..., \mathbf{x}_n^\nu] \in \mathbb{R}^{p_\nu \times n}$ be the data matrix of view ν, $\nu = 1, 2, ..., v$, and $Y = [\mathbf{y}_1, ..., \mathbf{y}_c] \in \mathbb{R}^{n \times c}$ is the normalized class indicator matrix, i.e. $Y_{ij} = 1/\sqrt{n_j}$ if the i-th data point belongs to the j-th class and $Y_{ij} = 0$ otherwise.

We try to minimize the residual of low rank regression model in each class and in each view. Loss function of multi-view low rank ridge regression can be proposed as in Eq.(4.1):

$$\begin{aligned} J_0 &= \sum_{\nu=1}^{v}\sum_{k=1}^{c}\{\|\mathbf{y}_k - (X_\nu^T\beta_k^\nu + f_k^\nu\mathbf{e})\|_2^2 + \lambda_\nu\|\beta_k^\nu\|_2^2\} \\ &= \sum_{\nu=1}^{v}\{\|Y - (X_\nu^T B^\nu + EF^\nu)\|_F^2 + \lambda_\nu\|B^\nu\|_F^2\} \end{aligned} \tag{4.1}$$

where projection matrix $B^\nu = [\beta_1^\nu, ..., \beta_c^\nu] \in \mathbb{R}^{p_\nu \times c}$, bias $F^\nu = diag(f_1^\nu, ..., f_c^\nu)$, $E = [\mathbf{e}, ..., \mathbf{e}] \in \mathbb{R}^{n\times c}$. $\mathbf{e}$ is a n-dimensional column vector with all elements equal to 1. λ_ν is the regularization parameter of view ν. Let's introduce low rank projection B^ν with rank s, $s < min(p_\nu, c)$,

$$\beta_k^\nu = A_\nu b_k, \quad \text{or} \quad B^\nu = A_\nu B, \tag{4.2}$$

where $A_\nu \in \mathbb{R}^{p_\nu \times s}$, and $B = (\mathbf{b}_1, ..., \mathbf{b}_c) \in \mathbb{R}^{s\times c}$. Therefore, the objective function Eq.(4.1) can be written as:

$$J_1 = \sum_{\nu=1}^{v}\{\|Y - (X_\nu^T A_\nu B + EF^\nu)\|_F^2 + \lambda_\nu\|A_\nu B\|_F^2\} \tag{4.3}$$

It is noteworthy that from Eq.(4.3), we can see that multi-view low-rank regression model is equivalent to do regression in the subspace of each view. Matrix A_ν is the subspace matrix of view ν. Matrix B is the shared regression parameter matrix of all views.

4.2.1 Closed form solution

We now present the closed form solution of the Multi-view Low Rank Regression. Before we talk about the closed form solution, we present Lemma 4 to simplify Eq.(4.3).

Lemma 4. *The bias f_k^ν can be solved and eliminated from J_1, which is thus simplified into*

$$J_1 = \sum_{\nu=1}^{v} \{\|Y^c - X_\nu^{cT} A_\nu B\|_F^2 + \lambda_\nu \|A_\nu B\|_F^2\} \tag{4.4}$$

where bias f_k^ν relates to B as

$$f_k^{\nu *} = \bar{y}_k - \bar{\mathbf{x}}_\nu^T A_\nu b_k^\nu \tag{4.5}$$

and $X_\nu^c = X_\nu - \bar{\mathbf{x}}\mathbf{e}^T$ is centered data matrix of view ν and $Y^c = Y - (\bar{y}_1, ..., \bar{y}_c)\mathbf{e}$ is centered class indicator matrix.

Proof. Taking derivative of Eq.(4.3) w.r.t. f_k^ν and setting it to zero, the optimal solution of f_k^ν is given as in Eq.(4.5), where $\bar{y}_k$ is a real number, $\bar{y}_k = \sum_{i=1}^n y_{ki}/n$, $\bar{\mathbf{x}}_\nu = \sum_{i=1}^n \mathbf{x}_i^\nu / n \in \mathbb{R}^{p_\nu \times 1}$. Substituting Eq.(4.5) into Eq.(4.3), we have Eq.(4.4). □

In the rest of this chapter, we focus on solving Eq.(4.4). For simplicity of notations, we drop c in X_ν^c and use X_ν to denote the centered X_ν. Similarly, we drop c in Y^c and use Y to denote the centered Y.

Now we present Theorem 3 to give the closed form solution of multi-view low-rank regression model.

Theorem 3. *The optimal solution of $J_1(\{A_\nu\}, B)$ is the following:*

1. $\{A_\nu\}$ is given by the optimal solution of the following problem:

$$\max_{\{A_\nu\}} Tr(G^{-1} H Y Y^T H^T) \tag{4.6}$$

where

$$G = G(\{A_\nu\}) \triangleq \sum_\nu A_\nu^T (X_\nu X_\nu^T + \lambda_\nu I) A_\nu, \tag{4.7}$$

$$H = H(\{A_\nu\}) \triangleq \sum_\nu A_\nu^T X_\nu \tag{4.8}$$

2. B is given by

$$B^* = G^{-1}H. \tag{4.9}$$

Proof. Taking derivative of Eq.(4.4) w.r.t. B, we have

$$\frac{\partial J}{\partial B} = -2\sum_{\nu} A_\nu^T X_\nu Y + 2\sum_{\nu} A_\nu^T X_\nu X_\nu^T A_\nu B + 2\lambda_\nu \sum_{\nu} A_\nu^T A_\nu B. \tag{4.10}$$

Setting Eq.(4.10) to zero, we have Eq.(4.9).

Substituting Eq.(4.9) in Eq.(4.4), we have

$$J = -\min_{\{A_\nu\}} \mathrm{Tr}(G^{-1}HYY^TH^T) \tag{4.11}$$

where $G = G(\{A_\nu\}) \triangleq \sum_\nu A_\nu^T(X_\nu X_\nu^T + \lambda_\nu I)A_\nu$,
$H = H(\{A_\nu\}) \triangleq \sum_\nu A_\nu^T X_\nu$. Eq.(4.11) is equivalent to Eq.(4.6). □

Furthermore, we present Theorem 4 to give the closed form solution for Eq.(4.6). Let

$$A = \begin{pmatrix} A_1 \\ A_2 \\ ... \\ A_v \end{pmatrix}, \qquad X = \begin{pmatrix} X_1 \\ X_2 \\ ... \\ X_v \end{pmatrix}, \tag{4.12}$$

$$S_b = XYY^TX^T, \tag{4.13}$$

$$S_t = diag(X_1X_1^T + \lambda_1 I, ..., X_vX_v^T + \lambda_v I), \tag{4.14}$$

Theorem 4. *Eq.(4.6) is equivalent to*

$$\max_A Tr[(A^TS_tA)^{-1}A^TS_bA], \tag{4.15}$$

where the optimal solution A^* is given by eigenvectors of $S_t^{-1}S_b$ that correspond to the s largest eigenvalues.

Algorithm 3 Multi-view low-rank regression

Input: Data matrix $X_\nu \in \mathbb{R}^{p_\nu \times n}$, class indicator matrix $Y \in \mathbb{R}^{n \times c}$, regularization weight parameter λ_ν, rank $s < c$, $\nu = 1, 2, ..., v$

Output: Matrix $A_\nu \in \mathbb{R}^{p_\nu \times s}$ and $B \in \mathbb{R}^{s \times c}$, $\nu = 1, 2, ..., v$

1: Compute S_b and S_t using Eq.(4.13) and Eq.(4.14)

2: Compute A_ν using the optimal solution of Eq.(4.15)

3: Compute B using Eq.(4.9)

Table 4.1: Multi-view datasets attributes.

Data	n	c	v	p_ν
MSRC	210	7	4	1302, 512, 100, 256
Caltech	1230	20	4	1302, 512, 100, 256
Cornell	195	5	3	107, 20, 15
Cora	2708	7	3	101, 180, 75

4.2.2 Algorithm

We present Algorithm 3 to summarize the steps of multi-view low-rank regression model. One of the advantages of our model is that it can be solved using closed-form solution directly. The input of this algorithm is (1) centered and normalized data matrix $X_\nu \in \mathbb{R}^{p_\nu \times n}$ from view ν, where $\nu = 1, 2, ..., v$, v is view number, p_ν is the dimension of view ν and n is sample number, (2) class indicator matrix $Y \in \mathbb{R}^{n \times c}$, (3) regularization weight parameter λ_ν, (4) rank s, which is less than the class number c. The output of this algorithm is matrix $A_\nu \in \mathbb{R}^{p_\nu \times s}$ and $B \in \mathbb{R}^{s \times c}$. We can compute S_b and S_t using Eq.(4.13) and Eq.(4.14). In step 2, we compute A, which is those eigenvectors of $S_t^{-1} S_b$ that correspond to the s largest eigenvalues. We should use Eq.(4.12) to restore A_ν from A. Finally, we compute B using Eq.(4.9).

4.3 Multi-view Full Rank Regression

Low-rank regression model has been proved to be an effective learning mechanism by exploring the low-rank structure of real life data. Will the multi-view low-rank regression model be able to capture the low-rank structure and improve the performance of a full-rank model? We will compare the performance of multi-view low-rank regression model with a full-rank model in experiment section.

In the case of multi-view full-rank regression, rank $s = c$, there is no constraint on B^ν in Eq.(4.1) and we will not use Eq.(4.2). To be specific, we will minimize the objective Eq.(4.4):

$$J_1 = \sum_{\nu=1}^{v} \{\|Y - X_\nu^T B^\nu\|_F^2 + \lambda_\nu \|B^\nu\|_F^2\} \tag{4.16}$$

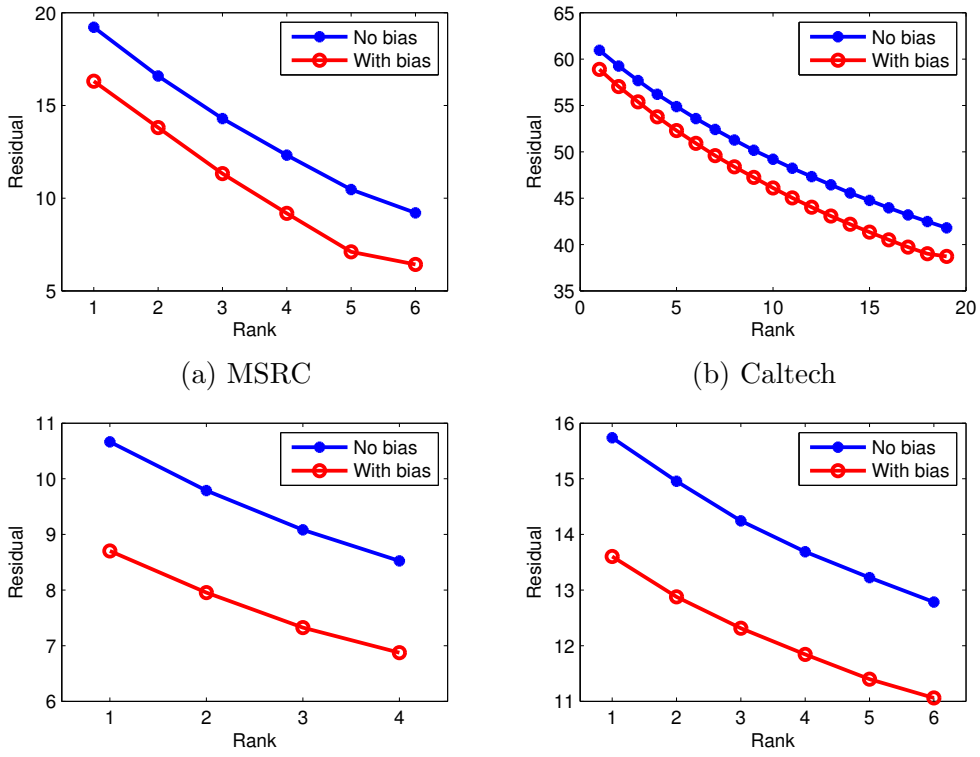
Eq.(4.16) can be solved using close form solution. Taking derivative of Eq.(4.16) w.r.t. B^ν and setting it to zero, the optimal solution of B^ν is given as

$$B^\nu = (X_\nu X_\nu^T + \lambda_\nu I)^{-1} X_\nu Y, \tag{4.17}$$

where $I \in \mathbb{R}^{p_\nu \times p_\nu}$ is an identity matrix.

4.4 Connections to other Multi-view work

Various multi-view learning models have been studied and all multi-view models are expected to have better performance than single view models. Existing multi-view approaches mainly are inspired from spectral clustering and subspace learning. de Sa [53] developed a spectral clustering algorithm for only two views by creating a bipartite graph based on the "minimizing-disagreement" idea. Zhou [54] developed a multi-view spectral clustering model via generalizing the single view normalized cut to the multi-view case. They try to find a cut which is close to be optimal on each single-view graph by exploiting a mixture of Markov chains associated with



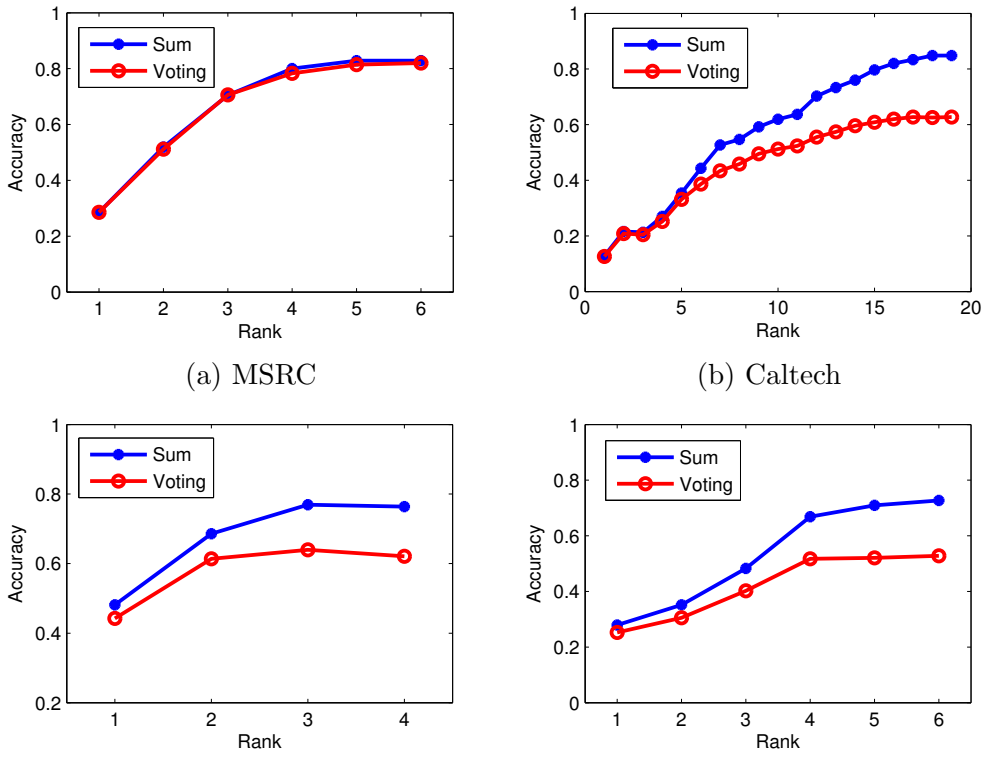
(a) MSRC (b) Caltech

(c) Cornell (d) Cora

Figure 4.1: Effect of regression bias in Eq.(4.3).

graphs of different views. Kumar [61] proposed a co-training flavour spectral clustering algorithm and use spectral embedding from one view to constrain the similarity graph used for the other view. Kumar [62] used the philosophy of co-regularization, which has been used in the past for semi-supervised learning problems, to make the clusterings in different views agree with each other.

Multi-view learning models from the point of view of subspace learning mainly try to find a subspace for each view and then develop a learning model across views in their subspaces. Canonical-Correlation Analysis (CCA) [63] was first used to study the correlation of two views in their respective subspaces. Hardoon [64] [65] designed an Kernel Canonical-Correlation Analysis to extract patterns from two views. Chaudhuri [66] proposed a CCA-based subspace multi-view learning approach to find



(a) MSRC (b) Caltech

(c) Cornell (d) Cora

Figure 4.2: Classification using different voting or sum methods.

a subspace such that the objects of different classes are well-separated and within-class distance is minimized. Greene [67] developed a Non-negative Matrix Factorization (NMF) [68] approach to effectively identify common patterns and reconcile between-view disagreements by combining data from multiple views.

The proposed multi-view low-rank regression model should be categorized into the class of subspace learning multi-view. The important contribution of this chapter is that we developed low-rank regression model to study multi-view problems. Surprisingly, there exists closed form solution to multi-view low-rank regression model.

4.5 Experiments

In this section, we perform extensive experiments on 4 multiple-view datasets. Through model learning, we systematically explore the best settings of regression bias, regularization weight parameter λ_ν and how to do classification using multi-view regression. We compare the classification accuracy of multi-view low-rank ridge regression with single-view regression, linear regression and full rank ridge regression.

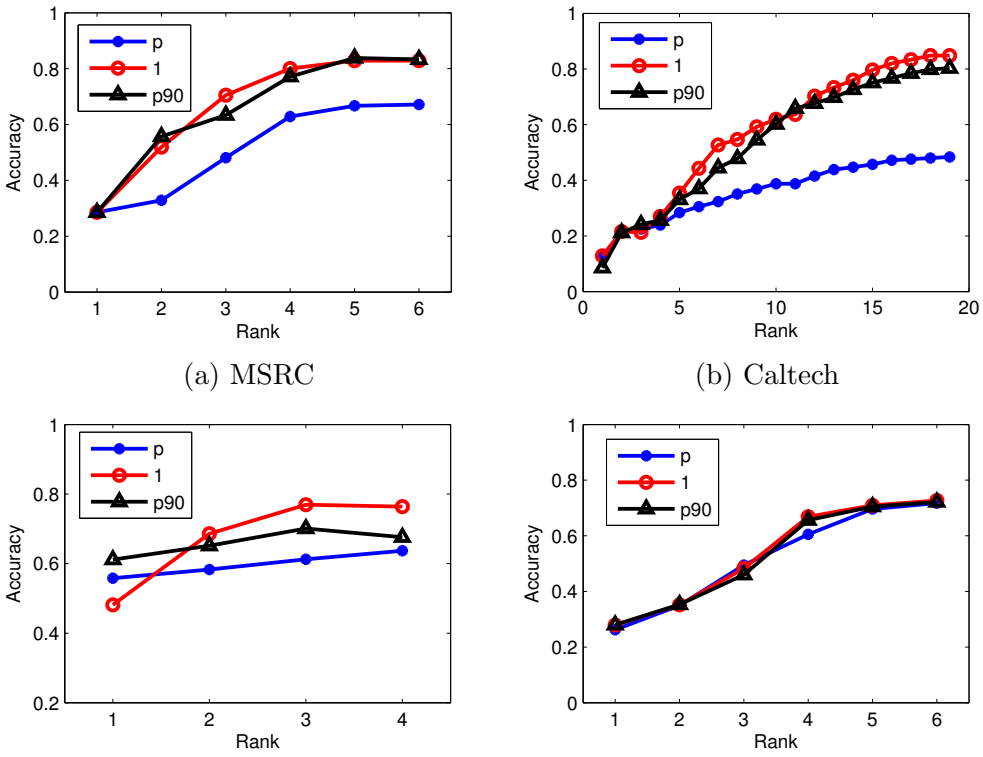
4.5.1 Datasets

Various multi-view datasets are used. These datasets include image datasets MSRC [24] and Caltech [23], website dataset Cornell [69] and scientific publication dataset Cora [70]. Cornell and Cora are downloaded from [71]. Summary of the datasets attributes are presented in Table 4.1, where n is sample number, c is class number, v is view number and p_ν lists the dimensions of different views.

MSRC is an image scene data, including trees, buildings, planes, cows, faces, cars and so on. It has 210 images from 7 classes. We extract different features from this data. The 4 views we used in this chapter are CENTRIST(1302 dimensions), GIST (512 dimensions), HOG (100 dimensions) and LBP (256 dimensions).

Caltech is a subset of Caltech 101 image data. It has images from 20 classes, including Faces, Leopards, Motorbikes, binocular, Brain, Camera, etc.. This data has 1230 images and 4 features are extracted from this data, including CENTRIST(1302 dimensions), GIST (512 dimensions), HOG (100 dimensions) and LBP (256 dimensions).

Cornell contains 195 documents over the 5 types (student, project, course, staff, faculty). There exists referral links among these documents. We use 3 views to describe the same document, including content view (107 dimensions), inbound-link view (20 dimensions) and outbound-link view (15 dimensions).



(a) MSRC (b) Caltech

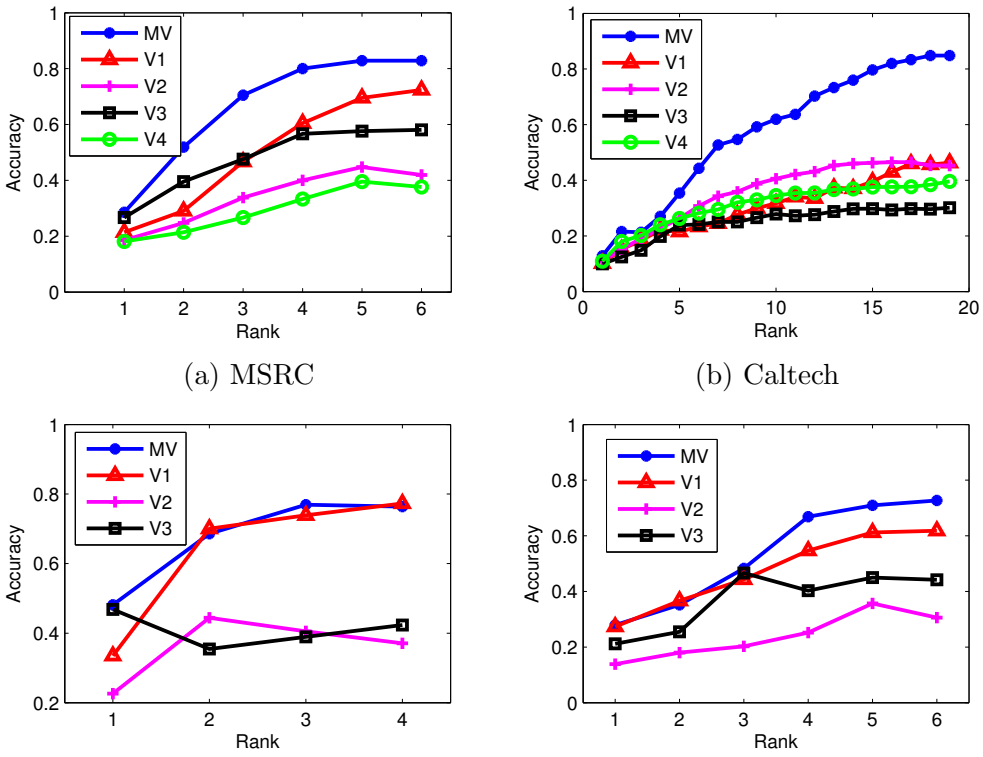
(c) Cornell (d) Cora

Figure 4.3: Regularization weight parameter λ_ν.

Cora consists of 2708 scientific publications classified into one of seven classes (Neural Networks, Rule Learning, Reinforcement Learning, Probabilistic Methods, Theory, Genetic Algorithms, Case Based). The citation network consists of links among those publications. The 3 views used in our experiments include content view (101 dimensions), inbound-link view (180 dimensions) and outbound-link view (75 dimensions).

4.5.2 Model learning

Through model learning, we systematically explore the best settings of regression bias, regularization weight parameter λ_ν and how to do classification using multi-view regression.



(a) MSRC (b) Caltech

(c) Cornell (d) Cora

Figure 4.4: Classification results of multi-view vs. single view data.

Effect of regression bias To validate that adding bias to regression will reduce fitting residual, Figure 4.1 compares the residual of class indicator matrix Y using two f_k^ν values: (1). using Eq.(4.5), denoted by "With bias" line (red circle line), (2)$f_k^\nu = 0$, denoted by "No bias" line (blue dot line). Residual r is defined as

$$r = \sum_{\nu=1}^{v} \|Y - X_\nu^T A_\nu B\|_F^2. \tag{4.18}$$

r is the summation of label matrix residuals over all views. Theoretically, adding bias F^ν could produce a more accurate fitting model, which means a model has smaller residual r. We examine this property by using rank $s = 1, ..., c-1$. As we can see from Figure 4.1, for all the 4 datasets, the residual using bias is always smaller than the residual without bias using all different ranks. In Figure 4.1a, 4.1c, and 4.1d, the

residual with bias ("With bias" line) is smaller than the residual without bias ("No bias" line). For MSRC data, the residual with bias is about 3 less than the residual without bias; for Caltech data, Figure 4.1b shows that the residual with bias is less than residual without bias; for Cornell data, the residual with bias is about 2 less on all rank numbers; for Cora data, the residual with bias is about 2 less on all rank numbers. In all, our results show that multi-view regression using bias could produce more accurate fitting models with less model residuals. In the following experiments, the default setting of all experiments is using bias.

Classification using regression In multi-view regression, there are different ways to do classification. For single-view low-rank regression [57],

$$\min_{A,B} \|Y - X^T AB\|_F^2 + \lambda \|AB\|_F^2, \tag{4.19}$$

where $A \in \mathbb{R}^{p\times s}$, $B \in \mathbb{R}^{s\times c}$ and AB is the low-rank regression parameter matrix, the following decision function is applied to classify a testing point $\mathbf{x} \in \mathbb{R}^{p\times 1}$ into one of the c classes,

$$\arg\max_{1\le j\le c} (\mathbf{y})_j, \tag{4.20}$$

where vector $\mathbf{y} = \mathbf{x}^T AB \in \mathbb{R}^{1\times c}$, class j corresponds to the index of the maximum value in vector $\mathbf{y}$.

In multi-view case, we predict a class using each view and then use majority voting to decide the final class. For example, for view ν, we use the following decision function to classify a testing point $\mathbf{x}_\nu \in \mathbb{R}^{p\times 1}$ into one of the c classes,

$$\arg\max_{1\le j\le c} (\mathbf{y}_\nu)_j, \tag{4.21}$$

where vector $\mathbf{y}_\nu = \mathbf{x}_\nu^T A_\nu B \in \mathbb{R}^{1\times c}$, $\mathbf{x}_\nu$ is the data vector of view ν, $\nu = 1, 2, ..., v$. Thus we predict a class label using every view. We have v predicted classes and apply

majority voting on v results. The class with most votes is assigned to this data point. If the top two classes get same number of votes, we assign them with 0.5 probability, etc.. We call this majority voting as "Voting" in Figure 4.2.

In this regression prediction problem, however, we can theoretically *derive* another voting method denoted as "Sum". Since our starting point is Eqs.(4.1-4.3), after obtaining A_ν and B through training, for a testing point $\mathbf{x}$, we learn $\mathbf{y}$ that minimizes the difference between label vector $\mathbf{y}$ and projected data of each views $\mathbf{x}_\nu^T A_\nu B$:

$$\min_{\mathbf{y}} \sum_{\nu=1}^{v} \|\mathbf{y} - \mathbf{x}_\nu^T A_\nu B\|_F^2. \tag{4.22}$$

It is obvious that the solution of Eq.(4.22) is given as

$$\mathbf{y} = (\sum_{\nu=1}^{v} \mathbf{x}^T A_\nu B)/v. \tag{4.23}$$

Once $\mathbf{y}$ is computed, we use Eq.(4.20) to obtain the class.

The classification accuracy using the two methods, Sum and Voting, is shown in Figure 4.2. As we can see from the results, for data Caltech, Cornell and Cora, Sum method has better results than Voting method obviously. Overall, the Sum voting method is better for regression based classification approach for multi-view regression. In the following experiments, the default setting of every experiment is using Sum method.

Regularization weight parameter λ_ν Regularization weight parameter λ_ν affects the regression model and classification accuracy directly. Many researchers tune this regularization weight parameter exponentially within a specific domain, such as from 10^{-5} to 10^5. It is very time consuming and misleading. In fact, regularization weight parameter λ_ν has direct contribution to the eigenvalues of $(X_\nu X_\nu^T + \lambda_\nu I)$, as shown in Eq.(4.7). A large λ_ν could change the distribution of eigenvalues of $(X_\nu X_\nu^T +$

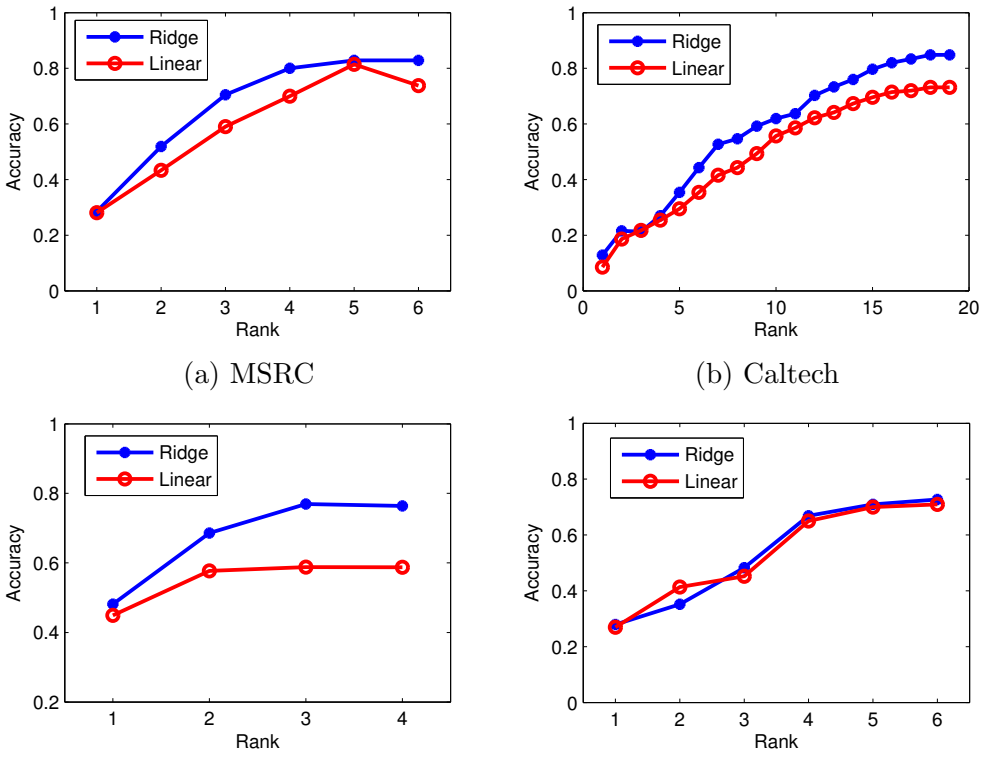
$\lambda_\nu I$) significantly. While a small λ_ν preserves the original eigenvalues distribution of $X_\nu X_\nu^T$. Thus, we constrain λ_ν to be the following 3 cases:

1. The summation for all the eigenvalues of $X_\nu X_\nu^T$. This will change the distribution of eigenvalues of $(X_\nu X_\nu^T + \lambda_\nu I)$ more significantly. Since X_ν is normalized row-wisely, $\lambda_\nu = \mathrm{Tr}(X_\nu X_\nu^T) = p_\nu$, where p_ν is dimension of view ν. In Figure 4.3, result using this method is denoted as "p".
2. The average of all the eigenvalues of $X_\nu X_\nu^T$. So $\lambda_\nu = \mathrm{Tr}(X_\nu X_\nu^T)/p_\nu = 1$, where p_ν is dimension of view ν. In Figure 4.3, result using this method is denoted as "1".
3. The 90%th largest eigenvalue. For example, if $X_\nu X_\nu^T$ has 200 non-zero eigenvalues sorted from large to small, we let λ_ν be the $90\% \times 200 = 180$th eigenvalue. This will change the distribution of eigenvalues of $(X_\nu X_\nu^T + \lambda_\nu I)$ slightly and still preserve the original eigenvalue distribution of $X_\nu X_\nu^T$. In Figure 4.3, result using this method is denoted as "p90".

Figure 4.3a shows that, for MSRC data, $\lambda_\nu = 1$ and "p90" performs better than using the summation of all eigenvalues ($\lambda_\nu = p_\nu$). In Figure 4.3b, $\lambda_\nu = 1$ can beat "p90" and $\lambda_\nu = p_\nu$. In Figure 4.3c, $\lambda_\nu = 1$ also has the best accuracy for rank $s = 2, 3, 4$. For data Cora, using different λ_ν does not affect accuracy too much. Over all, we choose λ_ν as the average of all eigenvalues of $X_\nu X_\nu^T$, which is $\lambda_\nu = 1$. In the following experiments, the default setting of every experiment is using $\lambda_\nu = 1$.

4.5.3 Comparison with single view

Multi-view regression uses data or information from multiple channels, such as different image features, both webpage citations view and contents view. Generally, we expect that multi-view regression can produce better results by exploiting information from multiple views. In this part, we compare multi-view low-rank regression

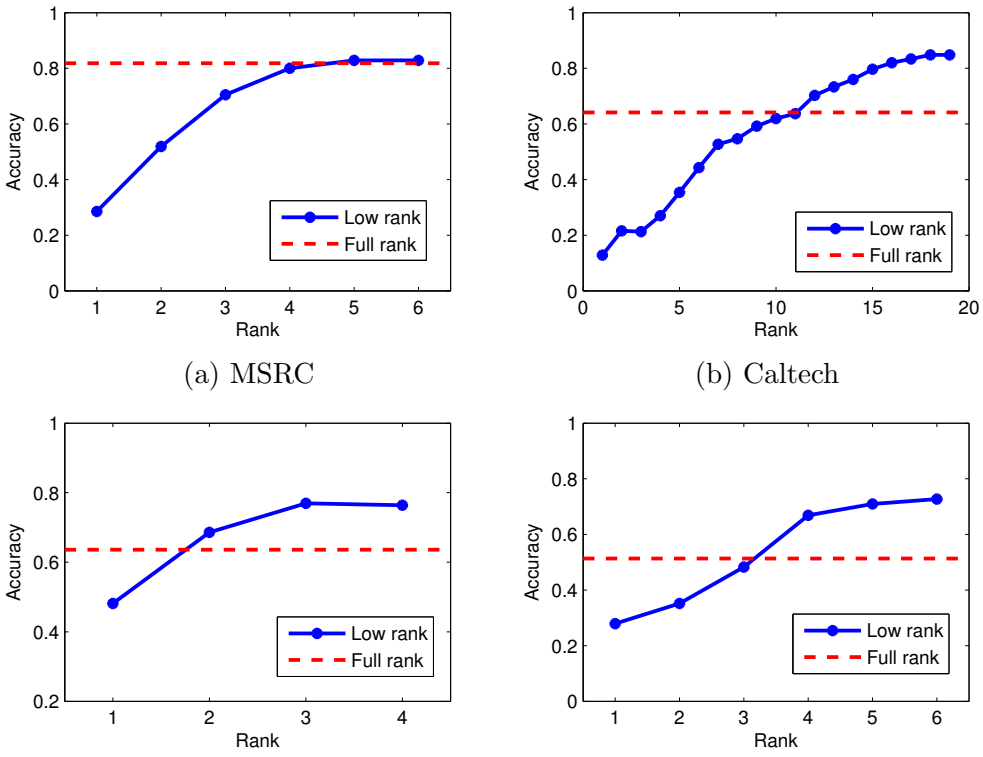


(a) MSRC (b) Caltech

(c) Cornell (d) Cora

Figure 4.5: Comparison of ridge regression and linear regression.

with single-view low-rank regression (see [57]). Figure 4.4 shows that multi-view low-rank regression produces better classification accuracy than single-view regression for different ranks (rank s is from 1 to $c-1$). "MV" denotes multi-view accuracy, "V1", "V2", ..., denote the accuracy using different single view. For example, Figure 4.4a shows that, for data MSRC, multi-view regression has much higher accuracy than all single-view low-rank regression when rank $s = 2, 3, 4, 5, 6$. Figure 4.4b shows that, when rank $s > 4$, multi-view regression has much higher accuracy than all the four single views. In Figure 4.4c, view "V1" has very good accuracy, but multi-view regression has better results than view "V1" when $s = 1, 3$. In Figure 4.4d, multi-view outperforms single-view when $s = 4, 5, 6$.



(a) MSRC (b) Caltech

(c) Cornell (d) Cora

Figure 4.6: Comparison of low-rank and full-rank.

4.5.4 Comparison of ridge regression and linear regression

Linear regression (when $\lambda_\nu = 0$) and ridge regression (when $\lambda_\nu \neq 0$) are closely related. Previous research [59] [57] shows that ridge regression will have better performance than linear regression. However, all existing work is based on single view. Does multi-view ridge regression produce better results than multi-view linear regression? We will examine the performance of multi-view linear regression and ridge regression on the 4 multi-view data with respect to different ranks. We can get linear regression by simply setting $\lambda_\nu = 0$ in our existing multi-view ridge regression model. Figure 4.5 shows that multi-view low-rank **ridge** regression ("Ridge" line in the figure) produces better classification accuracy than multi-view low-rank **linear** regression (when $\lambda_\nu = 0$, "Linear" line in the figure) in datasets MSRC, Caltech

and Cornell. For dataset Cora, ridge regression get slightly better results than linear regression when rank $s = 3, 4, 5, 6$.

4.5.5 Comparison of low-rank and full-rank

In real life, low-rank reveals the underlying structure of datasets and removes the noise and redundant information in the datasets. Low-rank regression model has been proved to be an effective learning mechanism by exploring the low-rank structure of real life data [55] [56] [57]. For full-rank regression, there is no constraint on B^{ν} in Eq.(4.1). We minimize the objective function of full-rank regression Eq. (4.16) and use the closed-form optimal solution given by Eq.(4.17) to solve the full-rank objective.

Figure 4.6 compares classification accuracy using low-rank multi-view regression and full-rank multi-view regression. The blue dot line is the low-rank classification accuracy for rank $s = 1, ..., c-1$, where c is class number. The red dash line is full-rank classification accuracy with rank $s = c$. The horizontal axis denotes rank of regression and the vertical axis denotes classification accuracy. As we can see, for all the 4 datasets, low-rank regression model can always beat full-rank regression model. For example, in Figure 4.6a, low-rank results with $s = 5$ and $s = 6$ have higher accuracy than full-rank with $s = 7$ (red dash line). In Figure 4.6b, low-rank results with $s = 11$ to $s = 19$ have higher accuracy than full-rank with $s = 20$. Figure 4.6c shows low-rank results with $s = 2, 3, 4$ have higher accuracy than full-rank with $s = 5$. Figure 4.6d shows low-rank results with $s = 4, 5, 6$ have higher accuracy than full-rank with $s = 7$.

4.6 Conclusion

In this chapter, we proposed a multi-view low-rank regression model. We provide a closed-form solution to multi-view low-rank regression model. Extensive experiments conducted on 4 multi-view datasets show that multi-view low rank regression outperforms full-rank regression counterpart and single-view counterpart in terms of classification accuracy.

CHAPTER 5

REGULARIZED SINGULAR VALUE DECOMPOSITION AND APPLICATION TO RECOMMENDER SYSTEM

5.1 Introduction

Singular value decomposition (SVD), its statistical form principal component analysis (PCA) and Karhunen-Loeve Transform in signal processing, are one of the most widely used mathematical formalism/decomposition in machine learning, data mining, pattern recognition, artificial intelligence, computer vision, signal processing, etc..

Mathematically, SVD can be seen as the best low-rank approximation to a rectangle matrix. The left and right singular vectors are mutually orthogonal, and provide orthogonal basis for row and column subspaces. When the data matrix are centered as in most statistical analysis, the singular vectors become eigenvectors of the covariance matrix and provide mutually uncorrelated/de-correlated subspaces which are much easier to use for statistical analysis. This form of SVD is generally referred to as PCA, and is widely used in statistics.

In its most simple form, SVD/PCA provides the most widely used dimension reduction for pattern analysis and data mining. SVD/PCA has numerous applications in engineering, biology, and social science [32] [72], such as handwritten zip code classification [73], human face recognition [74], gene expression data analysis [75], recommender system [76].

In recent developments of machine learning and data mining, regularization becomes an increasing trend. Adding a regularization term to the loss function can

increase the smoothness of the factor matrices and introduce more zero components to the factor matrices, such as sparse PCA [77] [78]. Sparse PCA has many applications in text mining, finance and gene data analysis [79] [80]. In this chapter, we present a regularized SVD (RSVD), present an efficient computational algorithm, and provide several theoretical analysis. We show that although the RSVD is a non-convex formulation, it has a global optimal closed-form solution. Finally, we apply RSVD to recommender system on four real life datasets. RSVD based recommender system outperforms the standard SVD based recommender system.

Notations. In this chapter, matrices are written in uppercase letters, such as X, Y. $\mathrm{Tr}(X)$ denotes the trace operation for matrix X.

5.2 Regularized SVD (RSVD)

Assume there is a matrix $X \in \mathbb{R}^{n\times m}$. Regularized SVD (RSVD) tries to find low-rank approximation using regularized factor matrices U and V. The objective function is proposed as

$$J_1 = \|X - UV^T\|_F^2 + \lambda\|U\|_F^2 + \lambda\|V\|_F^2, \tag{5.1}$$

where low-rank regularized factor matrices $U \in \mathbb{R}^{n\times k}$ and $V \in \mathbb{R}^{m\times k}$, k is the rank of regularized SVD. Minimizing Eq.(5.1) is a multi-variable problem. We will now present a faster Algorithm 4 to solve this problem.

Eq.(5.1) can be minimized in 2 steps:

A1. Fixing V, solve U. Take derivative of Eq.(5.1) with respect to U and set it to zero,

$$\frac{\partial J_1}{\partial U} = -XV + UV^TV + \lambda U = 0. \tag{5.2}$$

Thus we have Eq.(5.3):

$$U = XV(V^TV + \lambda I)^{-1}. \tag{5.3}$$

A2. Fixing U, solve V. Take derivative of Eq.(5.1) with respect to V and set it to zero,

$$\frac{\partial J_1}{\partial V} = -X^TU + VU^TU + \lambda V = 0. \tag{5.4}$$

Thus we can get the solution Eq.(5.5):

$$V = X^TU(U^TU + \lambda I)^{-1}. \tag{5.5}$$

It is easy to prove that function value J_1 is monotonically decreasing. To minimize objective function of Eq.(5.1), we propose an iterative Algorithm 4. We initialize V using a random matrix. Then we minimize Eq.(5.1) iteratively, until it converges. The converge speed is actually affected by the regularization weight parameter λ. In experiment section, we will show that RSVD converges faster than SVD ($\lambda = 0$).

Will the random initialization of matrix V in step 1 of Algorithm 4 affect the final solution? Is the solution of Algorithm 4 unique? Below, we present theoretical analysis and vigorously prove that there is a unique global solution and the above iterative algorithm converge to the global solution.

5.3 RSVD solution is in SVD subspace

Here we establish two important theoretical results: Theorems 5 and 6, which show RSVD solution is in SVD subspace.

The singular value decomposition (SVD) of X is given as

$$X = F\Sigma G^T, \tag{5.6}$$

Algorithm 4 Regularized SVD (RSVD)

Input: Data matrix $X \in \mathbb{R}^{n\times m}$, rank k, regularization weight parameter λ

Output: Factor matrices $U \in \mathbb{R}^{n\times k}$, $V \in \mathbb{R}^{m\times k}$

1: Initialize matrix V using a random matrix

2: **repeat**

3: Compute U using Eq.(5.3)

4: Compute V using Eq.(5.5)

5: **until** J_1 converges

where $F = (f_1, \cdots, f_r) \in \mathbb{R}^{n\times r}$ are the left singular vectors, $G = (g_1, \cdots, g_r) \in \mathbb{R}^{m\times r}$ are the right singular vectors, $\Sigma = \text{diag}(\sigma_1, ..., \sigma_r) \in \mathbb{R}^{r\times r}$ contains singular values, and r is the rank of X. $\sigma_1, ..., \sigma_r$ are sorted in decreasing order.

We now present Theorem 5 and 6 to show that RSVD solution is in subspace of SVD solution. Let V be the optimal solution of RSVD. Let the QR decomposition of $V \in \mathbb{R}^{m\times k}$ be

$$V = V_\perp \Omega, \tag{5.7}$$

where $V_\perp \in \mathbb{R}^{m\times k}$ is an orthonormal matrix and $\Omega \in \mathbb{R}^{k\times k}$ is an upper triangular matrix.

Theorem 5. *Matrix Ω in Eq.(5.7) is a diagonal matrix.*

Proof. Substituting Eq.(5.3) back into Eq.(5.1), we have a formulation of V only,

$$J_1(V) = \text{Tr}(X^T X - X^T X V (V^T V + \lambda I)^{-1} V^T + \lambda V^T V). \tag{5.8}$$

Using Eq.(5.7) and fixing $V_\perp$, we have

$$J_1(\Omega) = \text{Tr}(A - B\Omega(\Omega^T \Omega + \lambda I)^{-1} \Omega^T + \lambda \Omega^T \Omega), \tag{5.9}$$

where $A = X^T X, B = V_\perp^T X^T X V_\perp$ are independent of Ω. Let the eigen-decomposition of $\Omega^T\Omega = C\Lambda C^T, \Omega = \Lambda^{1/2} C^T$. Eq.(5.9) now becomes

$$J_1(\Lambda) = \mathrm{Tr}(A - B\Lambda^{1/2}(\Lambda + \lambda I)^{-1}\Lambda^{1/2} + \lambda\Lambda), \tag{5.10}$$

where C cancel out exactly. Thus J_1 is independent of C; J_1 depends on the eigenvalues of $\Omega^T\Omega$. For this reason, we can set $C = I$, $\Omega = \Lambda^{1/2}$ is a diagonal matrix. □

Theorem 6. *RSVD solution $V_\perp$ of Eq.(5.7) is in the subspace of SVD singular vectors G, as in Eq.(5.6).*

Proof. Using Eq.(5.7) and fixing Ω, Eq.(5.8) can be written as

$$J_1(V_\perp) = \mathrm{Tr}(A - V_\perp^T G\Sigma^2 G^T V_\perp D + E), \tag{5.11}$$

where $A = X^T X, D = \Omega(\Omega^T\Omega + \lambda I)^{-1}\Omega^T, E = \lambda\Omega^T\Omega$ is independent of $V_\perp$.

We now show that

(L1) For any $V_\perp$, $J_1(V_\perp)$ has a lower bound J_b:

$$J_1(V_\perp) \geq J_b = \mathrm{Tr}(A - \Sigma^2 D + E), \tag{5.12}$$

and

(L2) the optimal $V_\perp^* = G$.

To prove (L2), we see that when $V_\perp^* = G$,

$$J_1(V_\perp^*) = \mathrm{Tr}(A - G^T G\Sigma^2 G^T G D\Sigma^2 D + E) = J_b, \tag{5.13}$$

i.e., $J_1(V_\perp)$ reaches the lowest possible value, the global minima. Thus $V_\perp^* = G$ is the global optimal solution.

To prove (L1) we use Von Neumann's trace inequality, which states that for any two matrices P, Q, with diagonal singular value matrix Λ_P and Λ_Q respectively,

$|\mathrm{Tr}(PQ)| \leq \mathrm{Tr}(\Lambda_P \Lambda_Q)$. In our case, $Q = D = \Omega(\Omega^T\Omega + \lambda I)^{-1}\Omega^T$ is already a non-negative diagonal matrix. $P = V_\perp^T G\Sigma^2 G^T V_\perp$, and P's singular values are $\Sigma^2 > 0$. Thus we have

$$\mathrm{Tr}(V_\perp^T G\Sigma^2 G^T V_\perp D) \leq \mathrm{Tr}(\Sigma^2 D). \tag{5.14}$$

Adding constant matrices A, E and notice the negative sign, the inequality Eq.(5.14) gives the lower bound Eq.(5.12). This proves (L1). □

5.4 Closed form solution of RSVD

The key results of this chapter is that although RSVD is non-convex, we can obtain the global optimal solution, as below.

Using Theorems 5 and 6, we now present the closed form solution of RSVD. Given Eq.(5.3) and Eq.(5.7), as long as we solve Ω, we can get the closed form solution of RSVD U and V. The closed form solution is presented in Theorem 7.

Theorem 7. *Let SVD of the input data X be $X = F\Sigma G^T$ as in Eq.(5.6). Let (U^*, V^*) be the global optimal solution of RSVD. We have*

$$U^* = F_k, V^* = G_k\Omega \tag{5.15}$$

where $F_k = (f_1, \cdots, f_k)$, $G_k = (g_1, \cdots, g_k)$, and $\Omega = diag(\omega_1, ..., \omega_k) \in \mathbb{R}^{k\times k}$,

$$\omega_i = \sqrt{(\sigma_i - \lambda)_+},, \ i = 1, \cdots, k \tag{5.16}$$

Proof. Substituting Eq.(5.7) back to Eq.(5.8) and using $G^T G = I$, we have

$$J_1(\Omega) = \mathrm{Tr}(A - \Sigma^2\Omega^2(\Omega^2 + \lambda I)^{-1} + \lambda\Omega^2), \tag{5.17}$$

where $A = G\Sigma^2 G^T$ is a constant independent of Ω. Noting that all the matrices are diagonal, we can minimize J_1 element-wisely with respect to ω_i, $i = 1, ..., k$. Taking the derivative of J_1 respect to ω_i and setting it to zero, we have

$$\omega_i^2 = (\sigma_i - \lambda)_+, \tag{5.18}$$

because $\omega_i \geq 0$. From this, we finally have Eq.(5.16). □

One consequence of Theorem 7 is that the choice of parameter λ become obvious: it should be closedly related to parameter k, the rank of U, V.

We should set λ such that $\{\omega_i\} > 0$ so that no columns of U, V are waisted.

Another point to make is that directly computing U, V from Algorithm 1 is generally faster than compute the SVD of X, because generally, k are much smaller than rank(X), thus computing full rank SVD of X is not necessary.

Computational complexity analysis. From Theorem 3, a single SVD computation can obtain the global solution. If we desire a strong regularization, we set λ large, and compute SVD upto the appropriate rank using Eq.(5.18). The computation complexity is $O[k(n+m)\min(n,m)]$. We may use Algorithm 1 to directly compute RSVD without computing SVD. Theoretically, this is faster than computing the SVD because the regularization term $(V^TV + \lambda I)^{-1}$ makes Algorithm 4 converge faster for larger regularization λ. The The computation complexity is $O(kmn)$. Inverting the $k \times k$ matrix $(V^TV + \lambda I)$ is fast since k is typically much smaller than $\min(n, m)$.

Numerical experiments are given below.

5.5 Application to Recommender Systems

Recommender system generally uses collaborative filtering [76]. This is often viewed as a dimensionality reduction problem and their best-performing algorithm is based on singular value decomposition (SVD) of a user ratings matrix. By exploiting

the latent structure (low rank) of user ratings, SVD approach eliminates the need for users to rate common items. In recent years, SVD approach has been widely used as an efficient collaborative filtering algorithm [81] [76] [82] [83] [84].

User-item rating matrix X generally is a very sparse matrix with only values 1,2,3,4,5. Zeros elements imply that matrix entry has not been filled because each user usually only rates a few items. Similarly, each item is only rated by a small subset of users. Thus recommender system is in essence of estimating missing values of the rating matrix.

Assume we have a user-item rating matrix $X \in \mathbb{R}^{n\times m}$, where n is the number of users and m is the number of items (i.e., movies). Some ratings in matrix X are missing. Let Ω be the set of i, j indexes that the matrix element has been set. Recommender system using SVD solves the following problem:

$$\min_{U,V} \|X - UV^T\|_\Omega^2, \tag{5.19}$$

with fixed rank k of U, V, where for any matrix A, $\|A\|_\Omega^2 = \sum_{(i,j)\in\Omega} A_{ij}^2$.

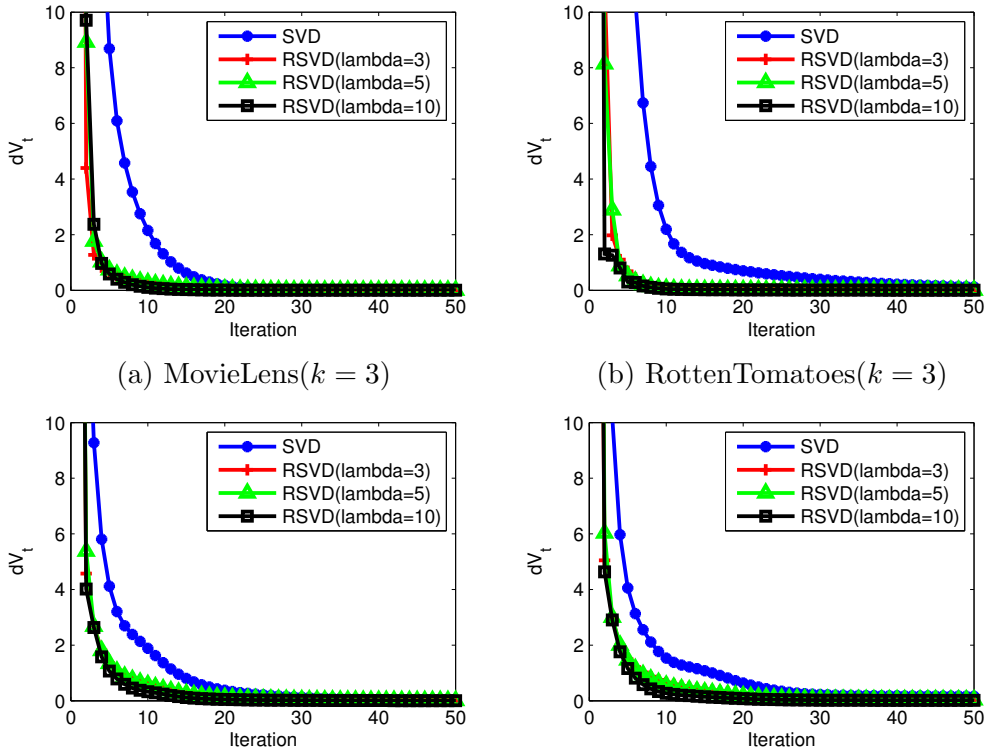
Low-rank U and V can expose the underlying latent structure. However, because X is sparse, U, V is forced to match a sparse structure and thus could overfit. Adding a regularization term will make U and V more smooth, and thus could reduce the overfitting. For this reason, we propose the regularized SVD recommender system as the following problem

$$\min_{U,V} \|X - UV^T\|_\Omega^2 + \lambda\|U\|_F^2 + \lambda\|V\|_F^2. \tag{5.20}$$

Both Eqs.(5.19,5.20) are solved by an EM-like algorithm [85] [86], which first fills the missing values with column or row averages, solving the low-rank reconstruction problem as the usual problem without missing values, and then update the missing values of X using the new SVD result. This is repeated until convergence. The RSVD algorithm presented above is used to solve Eqs.(5.19,5.20).

Table 5.1: Recommender system datasets.

Data	user (n)	item (m)
MovieLens	943	1682
RottenTomatoes	931	1274
Jester1	1731	100
Jester2	1706	100



(a) MovieLens($k = 3$) (b) RottenTomatoes($k = 3$)

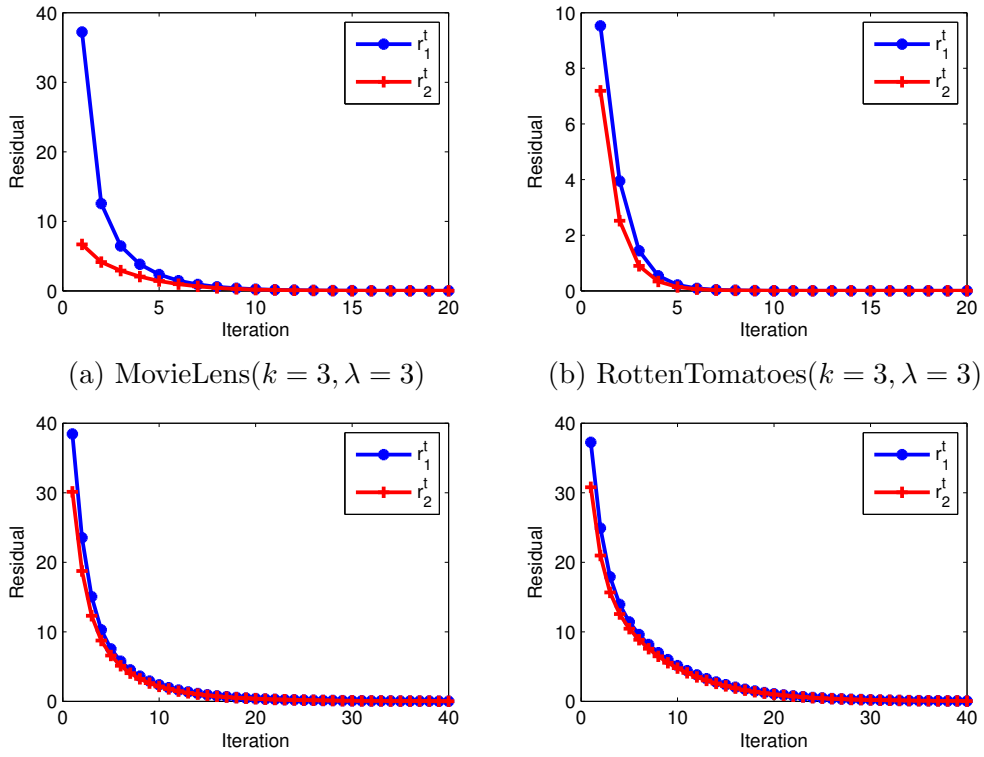
(c) Jester1($k = 14$) (d) Jester2($k = 14$)

Figure 5.1: RSVD convergence speed comparison at different λ, see Eq.(5.23).

5.6 Experiments

Here we compare recommender systems using the Regularized SVD of Eq.(5.20) and classical SVD of Eq.(5.19) on four datasets.

Datasets. Table 5.1 summarizes the user number n and item number m of the 4 datasets.



(a) MovieLens($k = 3, \lambda = 3$) (b) RottenTomatoes($k = 3, \lambda = 3$)

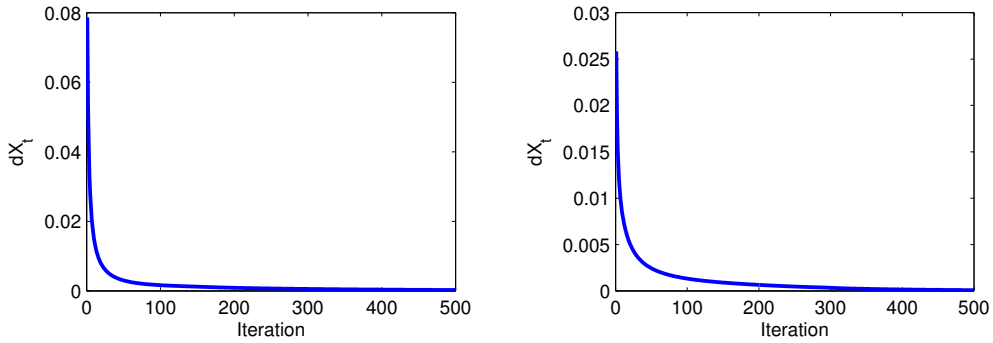
(c) Jester1($k = 14, \lambda = 3$) (d) Jester2($k = 14, \lambda = 3$)

Figure 5.2: RSVD share the same SVD subspace, see Eqs.(5.24,5.25).

MovieLens [87] [88] This data set consists of 100,000 ratings from 943 users on 1,682 movies. Each user has at least 20 ratings and the average number of ratings per user is 106.

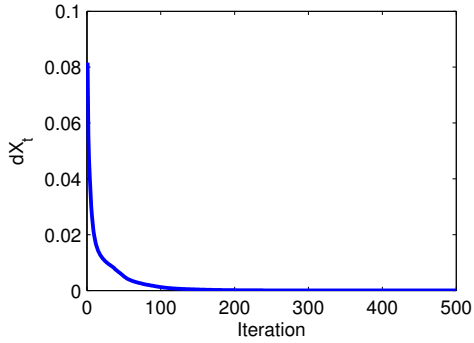
RottenTomatoes [87] [89] [90] This dataset contains 931 users and 1,274 artists. Each user has at least 2 movie ratings and the average number of ratings per user is 17.

Jester1 [81] Jester is an online Joke recommender system and it has 3 .zip files. Jester1 dataset contains 24,983 users and is the 1st .zip file of Jester data. In our experiments, we choose 1,731 users with each user having 40 or less joke ratings. The average number of ratings per user is 37.

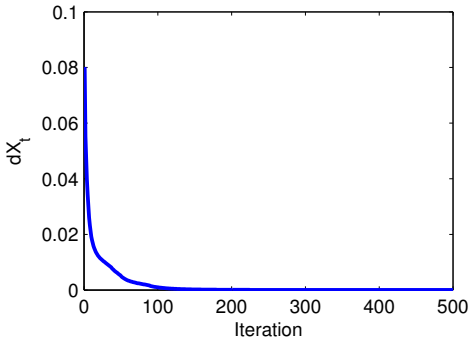


(a) MovieLens($k = 3, \lambda = 3$)

(b) RottenTomatoes($k = 3, \lambda = 3$)



(c) Jester1($k = 14, \lambda = 3$)



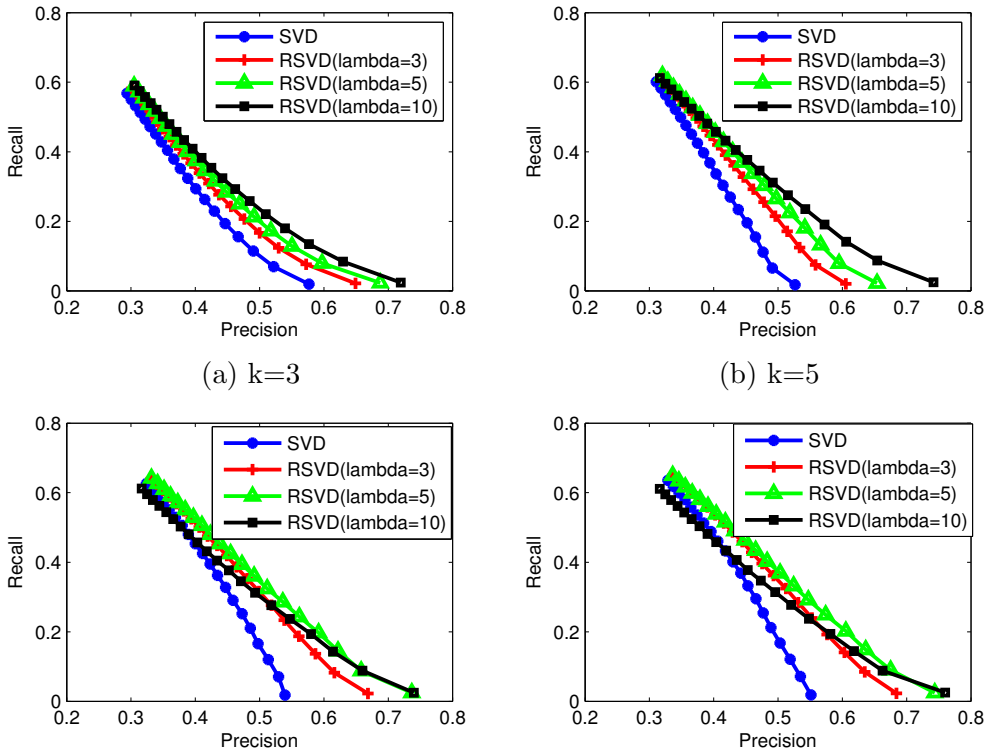
(d) Jester2($k = 14, \lambda = 3$)

Figure 5.3: Convergence of the solution to Recommender Systems of Eqs.(5.19,5.20) as the iteration of EM steps.

Jester2 [81] Jester2 dataset contains 23,500 users and is the 2nd .zip file of Jester data. In our experiments, we choose 1,706 users with each user having 40 or less joke ratings. The average number of ratings per user is 37.

5.6.1 Training data

Following standard approach, we convert all rated entries to 1 and all missing value entries remains zero. The evaluation methodology is: (1) construct training data by converting some 1s in the rating matrix into 0s, which is called "mask-out", (2) check if recommender algorithms can correctly recommend these masked-out ratings. Suppose we are given a set of user-item rating records, namely $X \in \mathbb{R}^{n \times m}$, where X



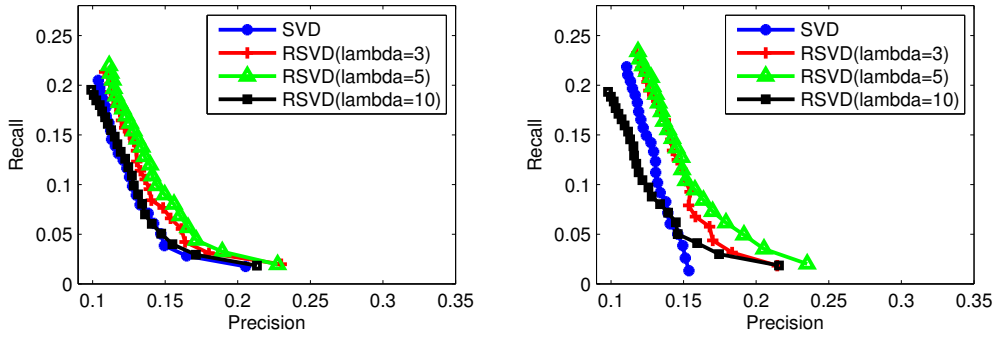
(a) k=3 (b) k=5

(c) k=7 (d) k=9

Figure 5.4: Precision and Recall curves on MovieLens.

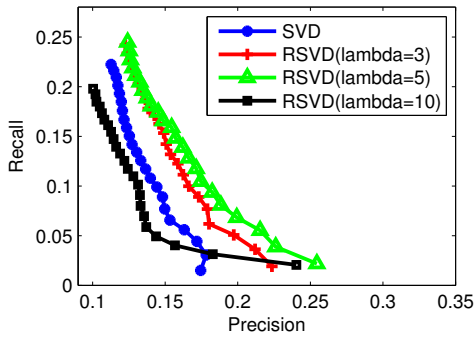
is the rating matrix, n is user number and m is item number. Each row of X denotes one user. To evaluate the performance of a recommender system algorithm, we need to know how accurate this algorithm can predict those 1s. We refer to the original data matrix as ground truth and mask out some ratings for some selected users. The **mask-out** process is as follows:

1. Find training users: those users with more than t ratings are selected as training users, where t is a threshold and t is a number related to the average ratings per user (m_{rating}). t controls the number of training users (n_{user}).
2. Mask out training ratings: for n_{user} selected training users, select n_{mask} ratings randomly per training user. In the user-item matrix X, we change those 1s into 0s.

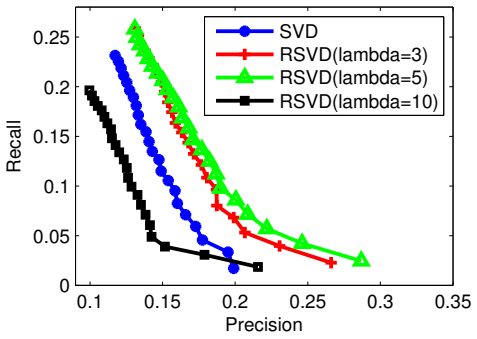


(a) k=3

(b) k=5



(c) k=7



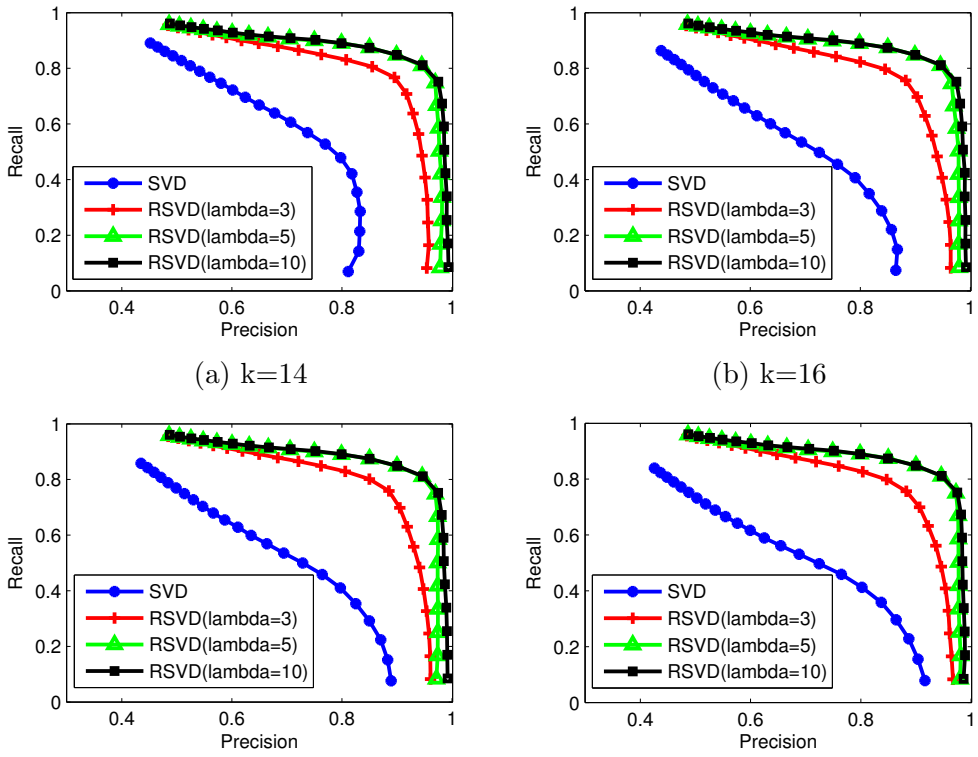
(d) k=9

Figure 5.5: Precision and Recall curves on RottenTomatoes.

Table 5.2: Training data parameter settings.

Data	t	n_{user}	m_{rating}	n_{mask}
MovieLens	100	361	106	90
RottenTomatoes	40	86	17	35
Jester1	37	803	37	35
Jester2	37	774	37	35

Table 5.2 shows the training data mask-out settings used in our experiments. It should be noted that these parameters are only one setting of constructing training datasets. Different settings will not make much difference, as long as we compare different recommender system algorithms on the same training dataset.



(a) k=14 (b) k=16

(c) k=18 (d) k=20

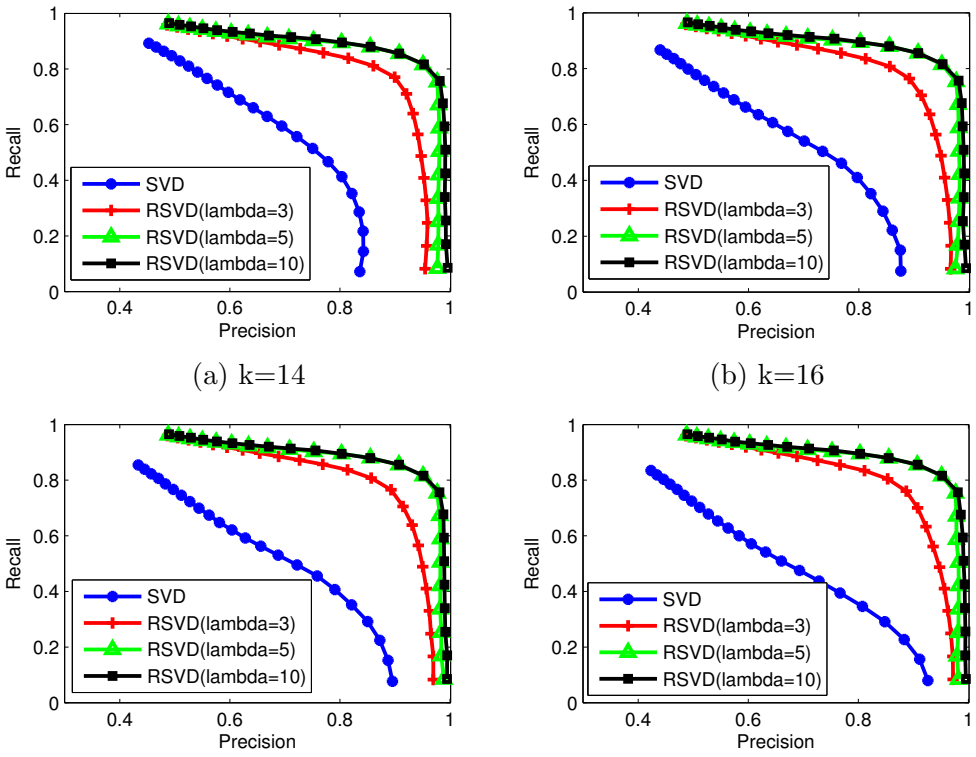
Figure 5.6: Precision and Recall curves on Jester1.

5.6.2 Top-N recommendation evaluation

To check if recommender algorithms can correctly recommend these masked-out ratings, we use Top-N recommendation evaluation method. Top-N recommendation is an algorithm to identify a set of N items that will be of interest to a certain users [91] [92] [84]. We use three metrics widely used in information retrieval community: recall, precision and F_1 measure. For each user, we first define three sets: $\mathbb{M}$, $\mathbb{T}$ and $\mathbb{H}$:

$\mathbb{M}$: Mask-out set. Size is n_{mask}. This set contains the ratings that are masked out(those values in data matrix X were changed from 1 to 0).

$\mathbb{T}$: Top-N set. Size is N. This set contains the N ratings that has the highest values (score) after using recommendation algorithm.



(a) k=14 (b) k=16

(c) k=18 (d) k=20

Figure 5.7: Precision and Recall curves on Jester2.

$\mathbb{H}$: Hit set. This set contains the ratings that appear both in $\mathbb{M}$ set and $\mathbb{T}$ set, $\mathbb{H} = \mathbb{M} \cap \mathbb{T}$.

Recall and precision are then defined as follows:

$$\text{Recall} = \frac{\text{size of set } \mathbb{H}}{\text{size of set } \mathbb{M}}, \text{Precision} = \frac{\text{size of set } \mathbb{H}}{\text{size of set } \mathbb{T}} \tag{5.21}$$

F_1 measure [93] combines recall and precision with an equal weight in the following form:

$$F_1 = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \tag{5.22}$$

We will get a pair of recall and precision using each N. In experiments, we use N from 1 to $2n_{mask}$, where n_{mask} is the number of ratings masked out per user. Thus, we can get a precision-recall curve in this way.

5.6.3 RSVD convergence speed comparison

Convergence speed is important for a faster iterative algorithm. We will compare the convergence speed of RSVD with iterative SVD algorithm ($\lambda = 0$). We define residual dV_t to measure the difference of V_t and V_{t-1} in two consecutive iterations:

$$dV_t = \|V_t - V_{t-1}\|_F, \tag{5.23}$$

where t is the iteration number of Algorithm 4. We compare RSVD with SVD ($\lambda = 0$) using different regularization weight parameter $\lambda = 3, 5, 10$. Figure 5.1 shows the dV_t decreases quickly along with iterations and RSVD converges faster than SVD.

5.6.4 RSVD share the same SVD subspace

From Theorem 6, we know that the solution of RSVD should be in the subspace of SVD solution. Formally, let U_t, V_t be the solution of RSVD after t iterations, F, G be the solution of SVD, $X = FG^T$. We now introduce Eq.(5.24) and Eq.(5.25) to measure the difference between U_t, V_t and F, G. r_1^t and r_2^t are defined as

$$r_1^t = \|U_t - FA_t\|_F^2, \tag{5.24}$$

$$r_2^t = \|V_t - GB_t\|_F^2. \tag{5.25}$$

In order to minimize r_1^t and r_2^t, the solution of A_t and B_t can be given as:

$$A_t = (F^T F)^{-1} F^T U_t, \tag{5.26}$$

$$B_t = (G^T G)^{-1} G^T V_t. \tag{5.27}$$

Substituting Eq.(5.26) and Eq.(5.27) back to Eq.(5.24) and Eq.(5.25), we get the minimized residual r_1^t and r_2^t. If r_1^t and r_2^t are equal to 0, it means that RSVD solution U_t and V_t share the same subspace as SVD solution F and G. Figure 5.2 shows residual r_2^t and r_3^t converges to 0 after a few iterations.

5.6.5 Convergence of recommender system solution

Solutions to the recommender systems Eqs.(5.19,5.20) converge. The EM-like algorithm has been shown effective in solving recommender systems [85] [86] [83] . We show the solution $(X_t)_\Omega$ converges after t iterations of EM-like iterations by using the difference,

$$dX_t = \frac{1}{\sqrt{N_\Omega}} \|X_t - X_{t-1}\|_\Omega. \tag{5.28}$$

where N_Ω is size of set Ω. Figure 5.3 shows the experiment result of dX_t. As we can see, for all the 4 datasets, the solution converges in about 100 to 200 iterations.

5.6.6 Precision-Recall Curve

In this part, we compare the precision and recall of RSVD and SVD using different rank k and regularization weight parameter λ. We use these k and λ settings because both RSVD and SVD models with these settings produce the best precision and recall. All the curves are the average results of 5 random run.

Figure 5.4 shows MovieLens data using SVD and RSVD with rank $k = 3, 5, 7, 9$. For each rank k, we compare SVD and RSVD with regularization weight parameter $\lambda = 3, 5, 10$. As we can see, for each rank k, RSVD performs better than SVD generally. Choosing λ properly could improve SVD algorithm and achieve the best precision and recall results.

Figure 5.5 shows RottenTomatoes data using SVD and RSVD with rank $k = 3, 5, 7, 9$. In all figures, RSVD with $\lambda = 5$ performs the best.

Figure 5.6 shows Jester1 data using SVD and RSVD with rank $k = 14, 16, 18, 20$. It is very easy to find that RSVD with $\lambda = 5, 10$ produce the best precision result for this data.

Table 5.3: F_1 measure (best values are in bold.).

Data	SVD	RSVD ($\lambda = 3$)	RSVD ($\lambda = 5$)	RSVD ($\lambda = 10$)
MovieLens (k=3)	0.3700	0.3850	0.3922	**0.4005**
MovieLens (k=5)	0.3875	0.4100	0.4199	**0.4232**
MovieLens (k=7)	0.4152	0.4391	**0.4439**	0.4231
MovieLens (k=9)	0.4220	0.4497	**0.4542**	0.4244
RottenTomatoes (k=3)	0.1220	0.1308	**0.1337**	0.1235
RottenTomatoes (k=5)	0.1302	0.1413	**0.1436**	0.1176
RottenTomatoes (k=7)	0.1315	0.1501	**0.1543**	0.1228
RottenTomatoes (k=9)	0.1422	0.1614	**0.1651**	0.1240
Jester1 (k=14)	0.6587	0.8241	**0.8668**	0.8665
Jester1 (k=16)	0.6201	0.8151	**0.8667**	0.8659
Jester1 (k=18)	0.6188	0.8213	**0.8672**	0.8666
Jester1 (k=20)	0.6077	0.8177	**0.8667**	0.8658
Jester2 (k=14)	0.6506	0.8305	**0.8730**	**0.8730**
Jester2 (k=16)	0.6261	0.8277	**0.8732**	0.8725
Jester2 (k=18)	0.6114	0.8288	**0.8729**	0.8723
Jester2 (k=20)	0.5908	0.8259	0.8721	**0.8722**

Figure 5.7 shows Jester2 data using SVD and RSVD with rank $k = 14, 16, 18, 20$. We can see from the results that RSVD with $\lambda = 5, 10$ produce the best precision result. As in Jester1 data, RSVD algorithm improves SVD significantly.

5.6.7 F_1 measure

F_1 measure combines precision and recall at the same time and can be used a good metric. F_1 measure is defined in Eq.(5.22). Since each N gives a pair of precision and recall, we use F_1 measure when $N = n_{mask}$ as the standard. Because $N = n_{mask}$, if all the masked-out ratings are predicted correctly, the size of set $\mathbb{H}$ can be exactly n_{mask}, which means recall is 1. F_1 measure ranges from 0 to 1. A higher F_1 measure (close to 1) means that an algorithm has better performance.

Table 5.3 shows the F_1 measure of the four datasets. Each row denotes a dataset with a specific rank k. The best F_1 measure is denoted in bold. As we can see, for all the datasets and ranks that we experimented, $\lambda = 5$ is a good setting that produces

the highest F_1 measure. In all, RSVD performs much better than SVD in terms of F_1 measure. In applications, we can test different λ and rank k setting to find the best setting for specific problems.

5.7 Conclusion

In conclusion, SVD is the mathematical basis of principal component analysis (PCA). We present a regularized SVD (RSVD), present an efficient computational algorithm, and provide several theoretical analysis. We show that although RSVD is non-convex, it has a closed-form global optimal solution. Finally, we apply regularized SVD to the application of recommender system and experimental results show that regularized SVD (RSVD) outperforms SVD significantly.

CHAPTER 6

MINIMAL SUPPORT VECTOR MACHINE

6.1 Introduction

Support Vector Machine (SVM) is an efficient classification approach, which finds a hyperplane to separate data from different classes. SVM has been widely used in object classification, face recognition, text categorization and so on. In most of these cases, SVM generalization performance either matches or is significantly better than that of competing methods [94].

Suppose we have n training samples from two classes $\{\mathbf{x}_i, y_i\}$, $i = 1, ..., n$, label indicator $y_i \in \{-1, 1\}$, $\mathbf{x}_i \in \mathbb{R}^{k\times 1}$, where k is data dimension. In linear separable case, suppose the hyperplane which separates the two classes is $\mathbf{w}^T\mathbf{x} + b = 0$, where $\mathbf{w} \in \mathbb{R}^{k\times 1}$ is normal to the hyperplane, $\mathbf{w}^T$ is the transpose of vector $\mathbf{w}$. Let d_+ (d_-) be the shortest distance from the separating hyperplane to the closest positive (negative) example. Define the margin of a separating hyperplane to be $d_+ + d_-$. Support Vector Machine finds such a separating hyperplane with the largest margin and all the training data satisfy the following constraints:

$$\mathbf{w}^T\mathbf{x}_i + b \geq +1 \text{ for } y_i = +1, \tag{6.1}$$

$$\mathbf{w}^T\mathbf{x}_i + b \leq -1 \text{ for } y_i = -1. \tag{6.2}$$

Combine the two equations into one:

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 \geq 0 \ \ \forall i. \tag{6.3}$$

Let the distance from origin of coordinate to the hyperplane $\mathbf{w}^T\mathbf{x} + b = 0$ be d_0, and let $d_0\mathbf{w}/\|\mathbf{w}\|$ be the point on the hyperplane that is closest to the origin, $\mathbf{w}/\|\mathbf{w}\|$ is a

unit vector that gives the direction perpendicular to the hyperplane. Since this point is on the hyperlane, we have $\mathbf{w}^T[d_0\mathbf{w}/\|\mathbf{w}\|] + b = 0$, thus $d_0 = |b|/\|\mathbf{w}\|$. Similarly, distance from origin to hyperplane $\mathbf{w}^T\mathbf{x}+b = -1$ is $|b+1|/\|\mathbf{w}\|$; distance from origin to hyperplane $\mathbf{w}^T\mathbf{x} + b = +1$ is $|b-1|/\|\mathbf{w}\|$. Hence, $d_+ = d_- = 1/\|\mathbf{w}\|$, and the margin is $2/\|\mathbf{w}\|$. Thus, for linear separable case, SVM objective is given as:

$$\min \frac{1}{2}\|\mathbf{w}\|^2, \tag{6.4}$$
$$\text{s.t. } y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 \geq 0 \ \ \forall i.$$

This can be solved using constrained optimization [94]. In testing, given a test data $\mathbf{x}$, we determine the class labels using $sign(\mathbf{w}^T\mathbf{x} + b)$.

When SVM is applied to non-separable data, non-negative slack variables ξ_i, $i = 1, ..., n$ are introduced to the constraints Eq.(6.1) and Eq.(6.2):

$$\mathbf{w}^T\mathbf{x}_i + b \geq +1 - \xi_i \text{ for } y_i = +1, \tag{6.5}$$
$$\mathbf{w}^T\mathbf{x}_i + b \leq -1 + \xi_i \text{ for } y_i = -1, \tag{6.6}$$
$$\xi_i \geq 0, \ \ \forall i. \tag{6.7}$$

Slack variables ξ_i measures training error. To minimize training errors and integrate slack variables into objective function, the non-separable SVM is given as:

$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i, \tag{6.8}$$
$$\text{s.t. } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i$$
$$\xi_i \geq 0, \ \ \forall i,$$

where C is a parameter that controls the weight of penalty to errors. Those training data that satisfy $y_i(\mathbf{w}^T\mathbf{x}_i + b) = 1 - \xi_i$, with $\xi_i \geq 0$, are called support vectors. We say that the constraints of support vectors are *active*. Support vectors decides the direction of the hyperplane.

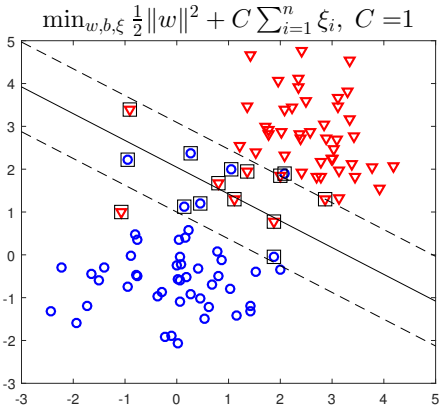
Nonlinear SVM is a generalized version of linear SVM. Suppose we have a mapping function that maps the data to some other Eculidean space $\mathcal{H}$, $\Phi : \mathbb{R}^{k\times 1} \rightarrow \mathcal{H}$. A kernel function using this mapping is $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$. Both in the training and testing process, we would only use the kernel function K and there is no need to know explicitly what Φ is.

Number of support vectors is a measure of generalization errors. Reducing number of support vectors can improve model prediction capability and classification accuracy can be improved. From the objective of Eq.(6.8), we can see that one way to reduce number of support vectors is to increase parameter C. However, we found that number of support vectors in Eq.(6.8) is not sensitive to C. In this work, we propose a Minimal SVM, which uses $L_{0.5}$ norm on slack variables. In Minimal SVM, number of support vectors is sensitive to C. On 7 binary classification tasks from 4 datasets, Minimal SVM further reduces the number of support vectors and increases the classification accuracy.

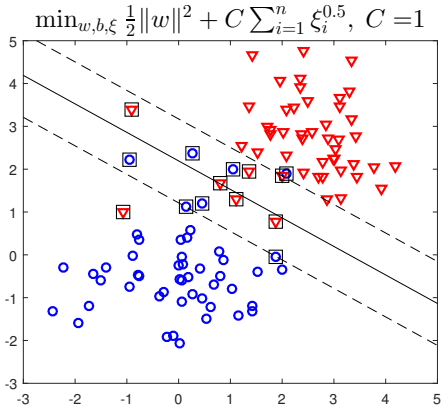
6.2 Motivation

In this section, we use a toy data set to show that number of support vectors in Eq.(6.8) is not sensitive to C. The toy data contains 100 2-dimensional random points from two classes, with 50 points in each class. Data points of each class are randomly generated by a normal distribution function. The two classes are non-separable.

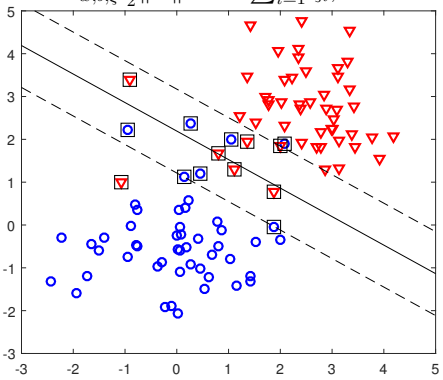
As we discussed in introduction, the hyperplane direction of SVM is determined by $\mathbf{w}$ and b. The width of margin is $2/\|\mathbf{w}\|$. Parameter C controls the weights of non-separable data errors. Figure 6.1a, 6.1c, 6.1e show the results using objective Eq.(6.8) when $C = 1$, 50, and 100. The solid black line is line $\mathbf{w}^T\mathbf{x} + b = 0$. The two dash black lines are $\mathbf{w}^T\mathbf{x} + b = -1$ and $\mathbf{w}^T\mathbf{x} + b = 1$. Two classes are denoted in blue circle and red triangle. Support vectors are those points with black squares.



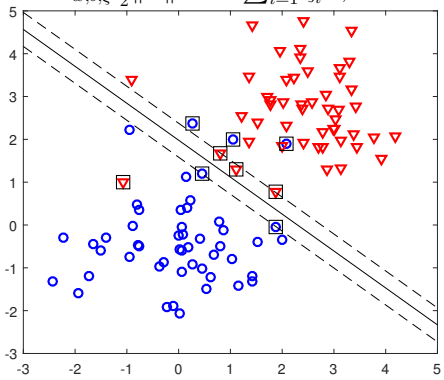
(a) $C = 1$, $nSV = 15$, margin $2/||\mathbf{w}|| = 1.7752$.



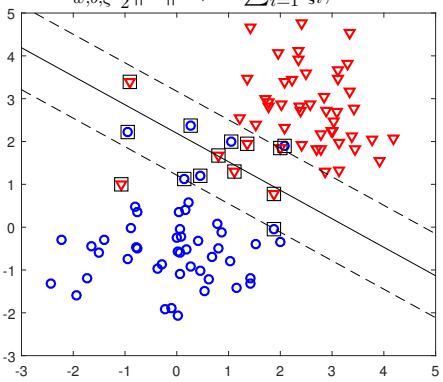
(b) $C = 1$, $nSV = 14$, margin $2/||\mathbf{w}|| = 1.3554$.



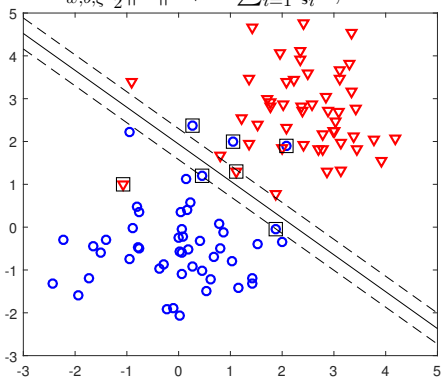
(c) $C = 50$, $nSV = 14$, margin $2/||\mathbf{w}|| = 1.6305$.



(d) $C = 50$, $nSV = 9$, margin $2/||\mathbf{w}|| = 0.5885$.



(e) $C = 100$, $nSV = 14$, margin $2/||\mathbf{w}|| = 1.6309$.



(f) $C = 100$, $nSV = 7$, margin $2/||\mathbf{w}|| = 0.5725$.

Figure 6.1: Comparison of SVM objective Eq.(6.8) and Eq.(6.9) on toy data (nSV is number of support vectors).

From Figure 6.1a, 6.1c, 6.1e, we can see that the number of support vectors can be further reduced and the number of support vectors is 15 when $C = 1$ and 14 when $C = 50, C = 100$. The width of margin is decreased when C increases. $2/\|\mathbf{w}\|$ is 1.7752 when $C = 1$, 1.6305 when $C = 50$, and 1.6309 when $C = 100$.

6.3 Minimal Support Vector Machine

L_p norm is a generalized version of L_1 and L_2 norm. When $0 \leq p \leq 1$, L_p norm introduces sparsity and has been used for feature selection [95]. In this chapter, we propose to solve the following Minimal Support Vector Machine (Minimal SVM) objective:

$$\begin{aligned} &\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i^p, \\ &\text{s.t. } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \\ &\quad \xi_i \geq 0 \ \ \forall i. \end{aligned} \tag{6.9}$$

When $p = 1$, Eq.(6.9) is the same as Eq.(6.8). When $p \to 0$, $\sum_i \xi_i^p$ approaches the number of nonzeros for $\xi_i, \forall i$. At small p, Eq.(6.9) will reduce number of nonzero ξ_i and the number of support vectors.

The primal Lagrangian of Eq.(6.9) is:

$$L_P = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i^p - \sum_i \alpha_i\{y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i\} - \sum_i \mu_i\xi_i, \tag{6.10}$$

where α_i and ξ_i are the Lagrange multipliers to enforce the positive constraints. The KKT conditions for the primal problem are given as:

$$\frac{\partial L_P}{\partial \mathbf{w}} = \mathbf{w} - \sum_i \alpha_i y_i \mathbf{x}_i = 0, \tag{6.11}$$

$$\frac{\partial L_P}{\partial b} = -\sum_i \alpha_i y_i = 0, \tag{6.12}$$

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i \geq 0, \tag{6.13}$$

$$\xi_i \geq 0, \tag{6.14}$$

$$\alpha_i \geq 0, \tag{6.15}$$

$$\mu_i \geq 0, \tag{6.16}$$

$$\alpha_i\{y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i\} = 0, \tag{6.17}$$

$$\xi_i(pC\xi_i^{p-1} - \alpha_i) = 0. \tag{6.18}$$

$\mathbf{x}_i^T$ is the transpose of row vector $\mathbf{x}_i$. Eq.(6.17, 6.18) are KKT complementarity conditions. Eq.(6.17) is the same as Eq.(55) in [94]. We can get Eq.(6.18) using $\partial L_P/\partial\xi_i = 0$ and $\mu_i\xi_i = 0$.

For ease of notation, we append b to vector $\mathbf{w}$ and append value 1 to $\mathbf{x}_i$

$$\mathbf{w}' = [\mathbf{w}, b] \tag{6.19}$$

$$\mathbf{x}_i' = [\mathbf{x}_i, 1] \tag{6.20}$$

Using Eqs.(6.14, 6.15, 6.17), Eq.(6.9) becomes a function with respect to vector $\mathbf{w}'$. When $\alpha_i > 0$, we have the following equation:

$$\xi_i = (1 - y_i\mathbf{w}'^T\mathbf{x}_i')_+, \tag{6.21}$$

where, for a number x, when $x > 0$, $(x)_+ = x$; when $x <= 0$, $(x)_+ = 0$. When $\alpha_i = 0$, from Eq.(6.18), we have $pC\xi_i^p = 0$, which implies $\xi_i = 0$.

Using Eq.(6.21), Eq.(6.9) becomes:

$$\min\frac{1}{2}\mathbf{w}'^T D\mathbf{w}' + C\sum_i(1 - y_i\mathbf{w}'^T\mathbf{x}_i')_+^p, \tag{6.22}$$

where $D \in \mathbb{R}^{(k+1)\times(k+1)}$ is an identity matrix with the last diagonal element $D(k+1, k+1)$ being 0. Eq.(6.22) can be solved using gradient descent with momentum [96].

Algorithm Since the derivative of function $(x)_+$ is not well defined when $x = 0$, we use the auxiliary function

$$(x)_+ = \lim_{s \to +\infty} \frac{1}{s} log(1 + \exp sx), \tag{6.23}$$

where s is a large number, for example, $s = 100$, $s = 200$.

The gradient of Eq.(6.22) is:

$$\nabla J(\mathbf{w}') = D\mathbf{w}' - pC \sum_i \frac{y_i m_i n_i^{p-1}}{1 + m_i} \mathbf{x}'_i, \tag{6.24}$$

where

$$m_i = \exp s(1 - y_i \mathbf{w}'^T \mathbf{x}'_i), \tag{6.25}$$

$$n_i = \frac{1}{s} log(1 + m_i). \tag{6.26}$$

Let $\eta > 0$ be the learning rate, $\varepsilon \in [0, 1]$ be the momentum coefficient, $\nabla J(\mathbf{w}'_t)$ be the gradient of Eq.(6.22) at iteration t.

$$\mathbf{v}_{t+1} = \varepsilon \mathbf{v}_t - \eta \nabla J(\mathbf{w}'_t), \tag{6.27}$$

$$\mathbf{w}'_{t+1} = \mathbf{w}'_t + \mathbf{v}_{t+1}, \tag{6.28}$$

$\mathbf{v}_t$ is initialized as vector of zeros. When optimal $\mathbf{w}'$ is found, we can get $\mathbf{w}$ and b using Eq.(6.19).

Algorithm 5 summarizes the steps to solve Eq.(6.9). Using the solution $\mathbf{w}$ and b of Algorithm 5, testing data $\mathbf{x}$ can be classified using $sign(\mathbf{w}^T\mathbf{x} + b)$. Support vectors are those points with positive ξ_i computed from Eq.(6.21).

Figure 6.1b, 6.1d, 6.1f are the results of applying objective Eq.(6.9) with $p = 0.5$ on the same toy data. We can see that the number of support vectors is reduced significantly when C increases from 1 to 100.

Algorithm 5 Gradient descent with Momentum to solve Eq.(6.9).

Input: Training data and label $\{\mathbf{x}_i, y_i\}$, $i = 1, ..., n$, parameter C, learning rate η, momentum coefficient ε

Output: $\mathbf{w}$, b

1: Initialize $\mathbf{w}$, b, $\mathbf{v}_0$

2: Form $\mathbf{w}'$ and $\mathbf{x}'_i$ using Eqs.(6.19, 6.20)

3: **while** Not converge **do**

4: Compute gradient using Eq.(6.24)

5: Compute $\mathbf{v}$ using Eq.(6.27)

6: Update $\mathbf{w}'$ using Eq.(6.28)

7: **end while**

Table 6.1: Data attributes.

Data	Dimension	Number of points in each class
MSRC	432	30
ATT	644	10
Binalpha	320	39
Caltech101	432	30

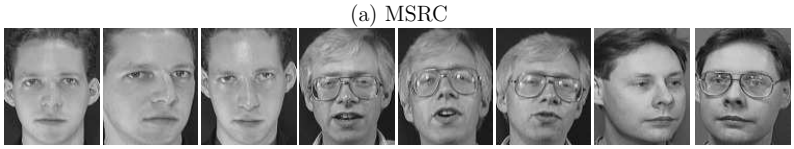
6.4 Experiments

In experiments, we select 7 binary classifcation tasks from 4 data sets as examples. We use $p = 0.5$ and study the convergence of Algorithm 5 and compare the classification performance of Minimal SVM and standard SVM.

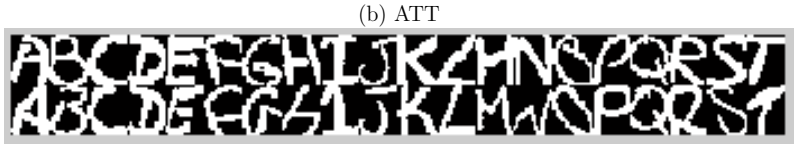
6.4.1 Data

Four image datasets are used in this experiment. Data attributes are summarized in Table 6.1. Example images are shown in Figure 6.2.

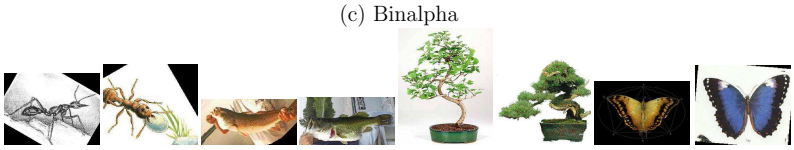
(a) MSRC



(b) ATT



(c) Binalpha



(d) Caltech101

Figure 6.2: Experiment example images.

Table 6.2: Experiment results ($p = 0.5$).

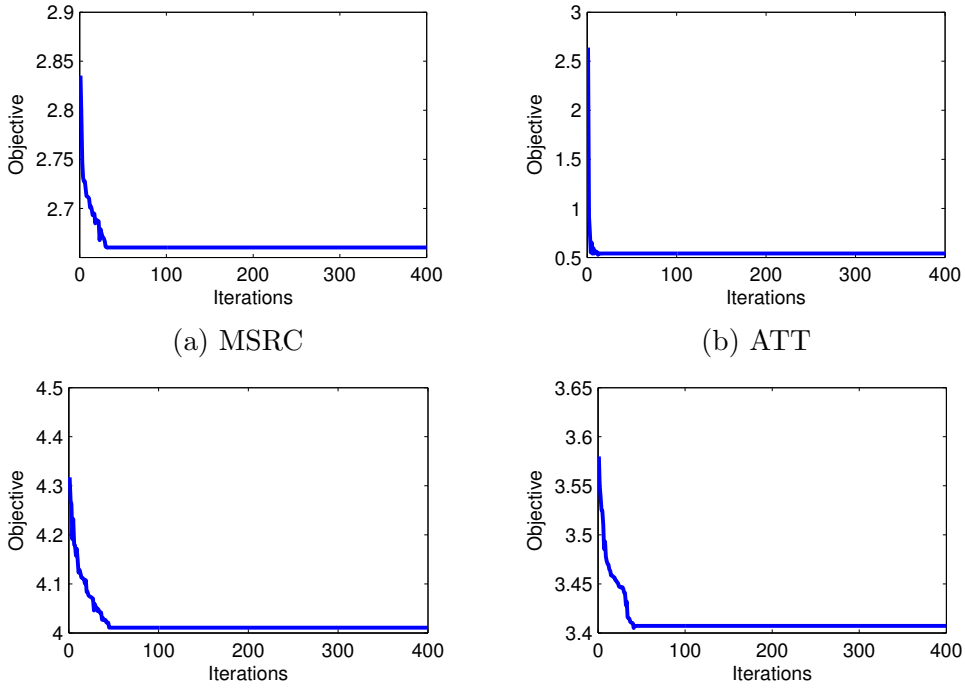
		MSRC	ATT	BinAlpha	Caltech101			
SVM	Test Acc	0.67	0.85	0.89	0.70	0.90	0.57	0.73
	Train Acc	**0.95**	**1.00**	**0.99**	**0.99**	**1.00**	**0.98**	**1.00**
	# SV	38.20	11.00	53.00	42.20	32.40	43.40	36.40
Minimal SVM	Test Acc	**0.72**	**0.90**	**0.90**	**0.73**	**0.92**	**0.58**	**0.75**
	Train Acc	**0.95**	**1.00**	**0.99**	0.97	**1.00**	0.95	0.98
	# SV	**22.40**	**2.00**	**33.40**	**31.80**	**18.40**	**30.80**	**17.80**
Angle θ		5.95	1.87	5.72	5.60	3.16	6.75	1.99
Dist d		0.13	0.06	0.12	0.11	0.09	0.14	0.04

MSRC[97] is an image scene data from MSRC data base v1, which includes tree, building, plane, cow, face, car and so on. 432-dimensional HOG feature is used in this chapter.

ATT [41] data contains 400 images of 40 persons, with 10 images for each person. The images has been resized to 28×23 pixels.

Binalpha data contains 26 binary hand-written alphabets. We use the 320-dimensional pixels feature.

Caltech101 [23] contains 101 object categories. We use the 432-dimensional HOG feature in this chapter.



(a) MSRC (b) ATT

(c) BinAlpha (d) Caltech101

Figure 6.3: Objective function Eq.(6.9) converges using Algorithm 5.

6.4.2 Convergence of Algorithm

Algorithm 5 is very efficient on the experiment datasets. Figure 6.3 shows Algorithm 5 on the four datasets converges in less than 50 iterations.

6.4.3 Evaluation

Table 6.2 shows the evaluation results using four data sets. Each column is a two-class classification experiments using standard SVM Eq.(6.8) solution $\mathbf{w}_{L1}$ and Minimal SVM Eq.(6.9) solution $\mathbf{w}_{L05}$ with $p = 0.5$. We compare the classification accuracy of testing, training and number of support vectors (# SV). Angle θ measures the angle degree between $\mathbf{w}_{L1}$ and $\mathbf{w}_{L05}$:

$$\theta = \arccos \frac{\mathbf{w}_{L1} \cdot \mathbf{w}_{L05}}{||\mathbf{w}_{L1}||||\mathbf{w}_{L05}||} \frac{180}{\pi}. \tag{6.29}$$

Distance d is the normalized Euclidean distance computed as:

$$d = \frac{||\mathbf{w}_{L1} - \mathbf{w}_{L05}||}{||\mathbf{w}_{L1}||}. \tag{6.30}$$

All experiments are the average of 5-fold cross validation results. The test accuracy and train accuracy number the is between 0 and 1, the larger the better. The number of support vectors are the smaller the better. Best results are in bold in Table 6.2. We can see that, Minimal SVM gives the best test classification on these two classes classification test and has much less support vectors compared to standard SVM. To further investigate the difference of $\mathbf{w}_{L1}$ and $\mathbf{w}_{L05}$, we found that the angle degree difference is between 1.87 to 6.75 degrees. The normalized Euclidean distance is between 0.04 and 0.14.

6.5 Conclusion

In this work, we proposed a Minimal SVM, which uses L_p norm on slack variables. We solve the objective using gradient descent with momentum by introducing a smoothing auxilary function. On 7 binary classification tasks, the proposed model further reduces the number of support vectors and increases the classification accuracy compared to standard SVM.

CHAPTER 7

CONCLUSIONS

In this dissertation, we made several advances in machine learning technologies for high dimensional data analysis, image data classification, recommender systems and classification algorithms. For high dimensional data analysis, we proposed two efficient Linear Discriminant Analysis (LDA) based methods, *kernel alignment inspired LDA* and *harmonic mean based LDA*, which can reduce high dimensional data to low dimensions, overcome the limitations of classical LDA and improve classification accuracy. For image data classification, we proposed a *multi-view low-rank regression model* which uses the correlations between different views of image data and imposes a low-rank constraints on multi-view data. For recommender system, we presented a *regularized SVD (RSVD) model for recommender system* to improve standard SVD based recommender system models. Finally, we proposed a Minimal Support Vector Machine (SVM) which uses L_p norm on slack variables. Minimal SVM further reduces the number of support vectors and increases the classification accuracy compared to standard SVM.

REFERENCES

[1] N. Cristianini, J. Shawe-taylor, A. Elisseeff, and J. S. Kandola, "On kernel target alignment," *Advances in neural information processing systems*, vol. 14, p. 367, 2002.

[2] N. Cristianini *et al.*, "Method of using kernel alignment to extract significant features from a large dataset," 2007, US Patent 7,299,213.

[3] X. Zhu, J. Kandola, Z. Ghahramani, and J. D. Lafferty, "Nonparametric transforms of graph kernels for semi-supervised learning," in *Advances in neural information processing systems*, 2004, pp. 1641–1648.

[4] S. C. Hoi, M. R. Lyu, and E. Y. Chang, "Learning the unified kernel machines for classification," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, pp. 187–196.

[5] A. Howard and T. Jebara, "Transformation learning via kernel alignment," in *Machine Learning and Applications, 2009. ICMLA'09. International Conference on*. IEEE, 2009, pp. 301–308.

[6] M. Cuturi, "Fast global alignment kernels," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 929–936.

[7] S. Zheng and C. Ding, "Kernel alignment inspired linear discriminant analysis," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer Berlin Heidelberg, 2014, pp. 401–416.

[8] H. Wang, C. Ding, and H. Huang, "Multi-label linear discriminant analysis," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 126–139.

[9] H. Zha, C. Ding, M. Gu, X. He, and H. Simon, "Spectral relaxation for K-means clustering," *Advances in Neural Information Processing Systems 14 (NIPS'01)*, pp. 1057–1064, 2001.

[10] C. Ding and X. He, "K-means clustering via principal component analysis," in *Proc of international conference on Machine learning (ICML 2004)*, 2004.

[11] H. Wang, S. Yan, D. Xu, X. Tang, and T. Huang, "Trace ratio vs. ratio trace for dimensionality reduction," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on.* IEEE, 2007, pp. 1–8.

[12] A. Edelman, T. A. Arias, and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM journal on Matrix Analysis and Applications*, vol. 20, no. 2, pp. 303–353, 1998.

[13] Y. Jia, F. Nie, and C. Zhang, "Trace ratio problem revisited," *Neural Networks, IEEE Transactions on*, vol. 20, no. 4, pp. 729–735, 2009.

[14] Y. Guo, T. Hastie, and R. Tibshirani, "Regularized linear discriminant analysis and its application in microarrays," *Biostatistics*, vol. 8, no. 1, pp. 86–100, 2007.

[15] H. Park, M. Jeon, and J. B. Rosen, "Lower dimensional representation of text data based on centroids and least squares," *BIT Numerical mathematics*, vol. 43, no. 2, pp. 427–448, 2003.

[16] J. Ye, "Characterization of a family of algorithms for generalized discriminant analysis on undersampled problems," in *Journal of Machine Learning Research*, 2005, pp. 483–502.

[17] J. Ye and S. Ji, "Discriminant analysis for dimensionality reduction: An overview of recent developments," *Biometrics: Theory, Methods, and Applications. Wiley-IEEE Press, New York*, 2010.

[18] H. Li, T. Jiang, and K. Zhang, "Efficient and robust feature extraction by maximum margin criterion," *Neural Networks, IEEE Transactions on*, vol. 17, no. 1, pp. 157–165, 2006.

[19] D. Kong and C. Ding, "A semi-definite positive linear discriminant analysis and its applications," in *2012 IEEE 12th International Conference on Data Mining (ICDM)*. IEEE, 2012, pp. 942–947.

[20] K. Yu, S. Yu, and V. Tresp, "Multi-label informed latent semantic indexing," in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2005, pp. 258–265.

[21] Y. Zhang and Z.-H. Zhou, "Multilabel dimensionality reduction via dependence maximization," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 4, no. 3, p. 14, 2010.

[22] S. Ji, L. Tang, S. Yu, and J. Ye, "Extracting shared subspace for multi-label classification," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008, pp. 381–389.

[23] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 59–70, 2007.

[24] Y. J. Lee and K. Grauman, "Foreground focus: Unsupervised learning from partially matching images," *International Journal of Computer Vision*, vol. 85, no. 2, pp. 143–166, 2009.

[25] T. Sim, S. Baker, and M. Bsat, "The cmu pose, illumination, and expression (pie) database of human faces," Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-01-02, January 2001.

[26] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[27] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. P. Vlahavas, "Multi-label classification of music into emotions." in *ISMIR*, vol. 8, 2008, pp. 325–330.

[28] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification." in *NIPS*, vol. 14, 2001, pp. 681–687.

[29] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[30] I. K. Fodor, "A survey of dimension reduction techniques," 2002.

[31] J. J. Dai, L. Lieu, and D. Rocke, "Dimension reduction for classification with gene expression microarray data," *Statistical applications in genetics and molecular biology*, vol. 5, no. 1, 2006.

[32] I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2005.

[33] K. Fukunaga, *Introduction to statistical pattern recognition*. Academic press, 2013.

[34] F. Nie, S. Xiang, and C. Zhang, "Neighborhood minmax projections." in *IJCAI*, 2007, pp. 993–998.

[35] L.-F. Chen, H.-Y. M. Liao, M.-T. Ko, J.-C. Lin, and G.-J. Yu, "A new lda-based face recognition system which can solve the small sample size problem," *Pattern recognition*, vol. 33, no. 10, pp. 1713–1726, 2000.

[36] D. B. Graham and N. M. Allinson, "Characterising virtual eigensignatures for general purpose face recognition," in *Face Recognition*. Springer, 1998, pp. 446–456.

[37] S. Zheng, F. Nie, C. Ding, and H. Huang, "A harmonic mean linear discriminant analysis for robust image classification," in *Tools with Artificial Intelligence (ICTAI), 2016 IEEE 28th International Conference on*. IEEE, 2016, pp. 402–409.

[38] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[39] F. Nie, S. Xiang, Y. Jia, and C. Zhang, "Semi-supervised orthogonal discriminant analysis via label propagation," *Pattern Recognition*, vol. 42, no. 11, pp. 2615–2627, 2009.

[40] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE transactions on pattern analysis and machine intelligence*, vol. 23, no. 6, pp. 643–660, 2001.

[41] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in *Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on*. IEEE, 1994, pp. 138–142.

[42] C. G. Snoek, M. Worring, J. C. Van Gemert, J.-M. Geusebroek, and A. W. Smeulders, "The challenge problem for automated detection of 101 semantic concepts in multimedia," in *Proceedings of the 14th ACM international conference on Multimedia*. ACM, 2006, pp. 421–430.

[43] D. Powers, "Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

[44] S. Zheng, Z.-Y. Shae, X. Zhang, H. Jamjoom, and L. Fong, "Analysis and modeling of social influence in high performance computing workloads," in *European Conference on Parallel Processing*. Springer Berlin Heidelberg, 2011, pp. 193–204.

[45] X. Zhang, Z.-Y. Shae, S. Zheng, and H. Jamjoom, "Virtual machine migration in an over-committed cloud," in *Network Operations and Management Symposium (NOMS), 2012 IEEE*. IEEE, 2012, pp. 196–203.

[46] D. Williams, S. Zheng, X. Zhang, and H. Jamjoom, "Tidewatch: Fingerprinting the cyclicality of big data workloads," in *INFOCOM, 2014 Proceedings IEEE*. IEEE, 2014, pp. 2031–2039.

[47] S. Zheng, A. Vishnu, and C. Ding, "Accelerating deep learning with shrinkage and recall," in *Parallel and Distributed Systems (ICPADS), 2016 IEEE 22nd International Conference on*. IEEE, 2016, pp. 963–970.

[48] S. Zheng, X. Cai, C. H. Ding, F. Nie, and H. Huang, "A closed form solution to multi-view low-rank regression." in *AAAI*, 2015, pp. 1973–1979.

[49] C. Shen, H. Li, and M. J. Brooks, "A convex programming approach to the trace quotient problem," in *Computer Vision–ACCV 2007*. Springer, 2007, pp. 227–235.

[50] F. Nie, S. Xiang, Y. Jia, C. Zhang, and S. Yan, "Trace ratio criterion for feature selection." in *AAAI*, 2008, pp. 671–676.

[51] W. Zhao, R. Chellappa, and P. J. Phillips, *Subspace linear discriminant analysis for face recognition*. Citeseer, 1999.

[52] S. Rüping and T. Scheffer, "Learning with multiple views," in *Proc. ICML Workshop on Learning with Multiple Views*, 2005.

[53] V. R. de Sa, "Spectral clustering with two views," in *ICML workshop on Learning with Multiple Views*, 2005.

[54] D. Zhou and C. J. Burges, "Spectral clustering and transductive learning with multiple views," in *Proceedings of the 24th International Conference on Machine Learning*. ACM, 2007, pp. 1159–1166.

[55] S. Xiang, Y. Zhu, X. Shen, and J. Ye, "Optimal exact least squares rank minimization," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data mining*. ACM, 2012, pp. 480–488.

[56] A. Evgeniou and M. Pontil, "Multi-task feature learning," *Advances in Neural Information Processing Systems*, vol. 19, p. 41, 2007.

[57] X. Cai, C. Ding, F. Nie, and H. Huang, "On the equivalent of low-rank linear regressions and linear discriminant analysis based regressions," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data mining*. ACM, 2013, pp. 1124–1132.

[58] G. A. Seber and A. J. Lee, *Linear regression analysis*. John Wiley & Sons, 2012, vol. 936.

[59] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[60] D. W. Marquaridt, "Generalized inverses, ridge regression, biased linear estimation, and nonlinear estimation," *Technometrics*, vol. 12, no. 3, pp. 591–612, 1970.

[61] A. Kumar and H. Daumé, "A co-training approach for multi-view spectral clustering," in *Proceedings of the 28th International Conference on Machine Learning*. ACM, 2011, pp. 393–400.

[62] A. Kumar, P. Rai, and H. Daume, "Co-regularized multi-view spectral clustering," in *Advances in Neural Information Processing Systems*, 2011, pp. 1413–1421.

[63] H. Hotelling, "Relations between two sets of variates," *Biometrika*, pp. 321–377, 1936.

[64] D. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural computation*, vol. 16, no. 12, pp. 2639–2664, 2004.

[65] D. R. Hardoon and J. Shawe-Taylor, "Convergence analysis of kernel canonical correlation analysis: theory and practice," *Machine Learning*, vol. 74, no. 1, pp. 23–38, 2009.

[66] K. Chaudhuri, S. M. Kakade, K. Livescu, and K. Sridharan, "Multi-view clustering via canonical correlation analysis," in *Proceedings of the 26th International Conference on Machine Learning*. ACM, 2009, pp. 129–136.

[67] D. Greene and P. Cunningham, "A matrix factorization approach for integrating multiple data views," in *Machine Learning and Knowledge Discovery in Databases*. Springer, 2009, pp. 423–438.

[68] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[69] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery, "Learning to construct knowledge bases from the world wide web," *Artificial Intelligence*, vol. 118, no. 1, pp. 69–113, 2000.

[70] A. McCallum, K. Nigam, J. Rennie, and K. Seymore, "A machine learning approach to building domain-specific search engines," in *IJCAI*, vol. 99. Citeseer, 1999, pp. 662–667.

[71] C. Grimal, "Multi-view datasets," http://lig-membres.imag.fr/grimal/data.html, 2014, [Online; accessed 11/17/2014].

[72] H. Zou, T. Hastie, and R. Tibshirani, "Sparse principal component analysis," *Journal of computational and graphical statistics*, vol. 15, no. 2, pp. 265–286, 2006.

[73] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer Series in Statistics New York, 2001, vol. 1.

[74] P. J. Hancock, A. M. Burton, and V. Bruce, "Face processing: Human perception and principal components analysis," *Memory & Cognition*, vol. 24, no. 1, pp. 26–40, 1996.

[75] O. Alter, P. O. Brown, and D. Botstein, "Singular value decomposition for genome-wide expression data processing and modeling," *Proceedings of the National Academy of Sciences*, vol. 97, no. 18, pp. 10 101–10 106, 2000.

[76] D. Billsus and M. J. Pazzani, "Learning collaborative information filters." in *ICML*, vol. 98, 1998, pp. 46–54.

[77] H. Shen and J. Z. Huang, "Sparse principal component analysis via regularized low rank matrix approximation," *Journal of multivariate analysis*, vol. 99, no. 6, pp. 1015–1034, 2008.

[78] Y. Guan and J. G. Dy, "Sparse probabilistic principal component analysis," in *International Conference on Artificial Intelligence and Statistics*, 2009, pp. 185–192.

[79] Y. Zhang, A. dAspremont, and L. El Ghaoui, "Sparse pca: Convex relaxations, algorithms and applications," in *Handbook on Semidefinite, Conic and Polynomial Optimization*. Springer, 2012, pp. 915–940.

[80] A. d'Aspremont, L. El Ghaoui, M. I. Jordan, and G. R. Lanckriet, "A direct formulation for sparse pca using semidefinite programming," *SIAM review*, vol. 49, no. 3, pp. 434–448, 2007.

[81] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative filtering algorithm," *Information Retrieval*, vol. 4, no. 2, pp. 133–151, 2001.

[82] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*. ACM, 2001, pp. 285–295.

[83] M. Kurucz, A. A. Benczúr, and K. Csalogány, "Methods for large scale svd with missing values," in *Proceedings of KDD Cup and Workshop*, vol. 12. Citeseer, 2007, pp. 31–38.

[84] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Application of dimensionality reduction in recommender system-a case study," DTIC Document, Tech. Rep., 2000.

[85] N. Srebro and T. Jaakkola, "Weighted low-rank approximations," in *ICML*, vol. 3, 2003, pp. 720–727.

[86] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[87] GroupLens, "Grouplens research group," http://www.grouplens.org, 2014.

[88] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999, pp. 230–237.

[89] IMDB, "Imdb website," http://www.imdb.com, 2014.

[90] RottenTomatoes, "Rotten tomatoes website," http://www.rottentomatoes.com, 2014.

[91] G. Karypis, "Evaluation of item-based top-n recommendation algorithms," in *Proceedings of the tenth international conference on Information and knowledge management*. ACM, 2001, pp. 247–254.

[92] M. Deshpande and G. Karypis, "Item-based top-n recommendation algorithms," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, pp. 143–177, 2004.

[93] Y. Yang and X. Liu, "A re-examination of text categorization methods," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999, pp. 42–49.

[94] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[95] M. Zhang, C. H. Ding, Y. Zhang, and F. Nie, "Feature selection at the discrete limit." in *AAAI*, 2014, pp. 1355–1361.

[96] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, "On the importance of initialization and momentum in deep learning." *ICML (3)*, vol. 28, pp. 1139–1147, 2013.

[97] J. Winn and N. Jojic, "Locus: Learning object classes with unsupervised segmentation," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 1. IEEE, 2005, pp. 756–763.

BIOGRAPHICAL STATEMENT

Shuai Zheng was born in Henan, China, in 1986. He received his Ph.D. degree in Computer Science from the University of Texas at Arlington in 2017. His main research interests are machine learning, data mining and cloud computing. Before coming to Texas, Shuai Zheng received his M.S. degree in Applied Mathematics and Computational Science from King Abdullah University of Science and Technology (KAUST), Saudi Arabia in 2011 and B.S. degree in Physics from Jilin University, China in 2009.