Tampere University

Kateryna Chumachenko

MULTI-VIEW SUBSPACE LEARNING FOR LARGE-SCALE MULTI-MODAL DATA ANALYSIS

Faculty of Information Technology and Communication Sciences
Master of Science Thesis
August 2019

ABSTRACT

Kateryna Chumachenko: MULTI-VIEW SUBSPACE LEARNING FOR LARGE-SCALE MULTI-MODAL DATA ANALYSIS
Master of Science Thesis
Tampere University
Master's Degree Programme in Information Technology
Major: Data Engineering and Machine Learning
August 2019

Dimensionality reduction methods play a big role within the modern machine learning techniques, and subspace learning is one of the common approaches to it. Although various methods have been proposed over the past years, many of them suffer from limitations related to the unimodality assumptions on the data and low speed in the cases of high-dimensional data (in linear formulations) or large datasets (in kernel-based formulations). In this work, several methods for overcoming these limitations are proposed.

In this thesis, the problem of the large-scale multi-modal data analysis for single- and multi-view data is discussed, and several extensions for Subclass Discriminant Analysis (SDA) are proposed. First, a Spectral Regression Subclass Discriminant Analysis method relying on the Graph Embedding-based formulation of SDA is proposed as a way to reduce the training time, and it is shown how the solution can be obtained efficiently, therefore reducing the computational requirements. Secondly, a novel multi-view formulation for Subclass Discriminant Analysis is proposed, allowing to extend it to data coming from multiple views. Besides, a speed-up approach for the multi-view formulation that allows reducing the computational requirements of the method is proposed. Linear and nonlinear kernel-based formulations are proposed for all the extensions.

Experiments are performed on nine single-view and nine multi-view datasets and the accuracy and speed of the proposed extensions are evaluated. Experimentally it is shown that the proposed approaches result in a significant reduction of the training time while providing competitive performance, as compared to other subspace-learning based methods.

Keywords: subspace learning, kernel methods, dimensionality reduction, spectral regression, subclass discriminant analysis

PREFACE

The work in this thesis was conducted in the Multimedia Research Group in the Department of Computing Sciences in the Faculty of Information Technology and Communication Sciences in Tampere University in 2019.

I would like to express my sincere gratitude to Prof. Moncef Gabbouj, Assoc. Prof. Alexandros Iosifidis and Dr. Jenni Raitoharju for giving me the opportunity to work as a research assistant within the group and providing constant guidance and support throughout the work process.

Besides, I would like to thank all the members of Multimedia Research Group for creating such a nice and motivational work atmosphere.

In addition, I would like to thank my family for their support throughout my studies and my life in general.

Tampere, 1st August 2019

Kateryna Chumachenko

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CDA	Clustering Discriminant Analysis
KCDA	Kernel Clustering Discriminant Analysis
KSDA	Kernel Subclass Discriminant Analysis
KSMFA	Kernel Subclass Marginal Fisher Analysis
LDA	Linear Discriminant Analysis
MFA	Marginal Fisher Analysis
MvMDA	Multi-view Modular Discriminant Analysis
NPT	Nonlinear Projection Trick
PCA	Principal Component Analysis
RBF	Radial Basis Function
SDA	Subclass Discriminant Analysis
SGE	Subclass Graph Embedding
SMFA	Subclass Marginal Fisher Analysis
SMvDA	Standard Multi-view Discriminant Analysis
SRDA	Spectral Regression Discriminant Analysis
fastSDA	Fast Subclass Discriminant Analysis
mvSDA	Multi-view Subclass Discriminant Analysis

1 INTRODUCTION

Availability of the large amounts of data in the modern world dictates the development of new technologies for processing and analysing it, giving rise to the field of machine learning. In the past years, machine learning has been applied to solve problems in many subject areas, including image processing [38, 61], audio and speech analysis [3, 20], human action recognition [17], etc. In many areas, such as face recognition or object detection, machine learning-based methods are the dominant approach to solving problems [45, 47, 48, 51]. Although the presence of rich data from different sources allows improving the accuracy of the algorithms, it also raises issues related to their computational requirements. In this thesis, we study a subfield of machine learning, namely, subspace learning, and propose several extensions for speeding-up and improving the accuracy of existing methods.

The goal of machine learning is to estimate the parameters of a mathematical model to perform a specific task without using explicit instructions, but relying on the patterns in the data, from which it learns. The data, in this case, is represented by a matrix or a tensor, i.e., multiple samples described by one or multiple features. Intuitively it might seem that a larger amount of features should result in better accuracy of the model, which is true only up to a certain point: when the dimensionality of data becomes too high, the accuracy of the model drops, as the data representation becomes sparse, i.e., there are not enough samples to capture enough possible combinations of the features. This phenomenon is known as the 'curse of dimensionality'. Another limitation that comes with high dimensionality is the high computational complexity that results in low processing speed. These issues resulted in the formation of a research area referred to as dimensionality reduction.

Dimensionality reduction methods aim to find such a representation of data that would result in a lower dimensionality than that of original data while preserving the 'meaningfulness' of data. The 'meaningfulness' can be defined by different criteria, e.g., some methods seek representation with the highest variance, while others a representation, where data belonging to different classes lies far from each other, while classes are compact.

Machine learning methods can be divided into supervised and unsupervised ones. Supervised machine learning, also referred to as 'learning with a teacher', relies on the ground truth labels present for all data samples in the training set. Examples of supervised machine learning problems are regression and classification. Unsupervised learning methods do not rely on the ground truth labels of data but try to find certain patterns

in the data. Clustering is one of the examples of unsupervised learning. In this thesis, we focus on supervised dimensionality reduction methods that aim at classification problems.

One of the common dimensionality reduction approaches is subspace learning, which aims to find a projection subspace of a lower dimensionality for the data while ensuring that a certain criterion holds for the data projected onto the subspace. Examples of subspace learning methods include Principal Component Analysis (PCA) [16], Linear Discriminant Analysis (LDA) [57, 60], Subclass Discriminant Analysis (SDA) [66], etc. Linear Discriminant Analysis is one of the most well-known subspace learning methods designed primarily for classification problems. LDA defines an optimal projection space as the one where the distances between the different class means are maximal, while the classes are compact. The solution is based on the assumption that each class follows a unimodal Gaussian distribution. Several limitations come from this assumption:

- limited dimensionality of the learned space, as it is limited by the rank of the between-class scatter matrix, which is bounded by the number of classes - 1
- poor performance on multi-modal datasets, as LDA assumes that data of each class follows a unimodal Gaussian distribution
- high computational complexity in high-dimensional datasets, as the solution is given by the eigendecomposition of a large matrix.

Multiple approaches to overcoming these limitations have been proposed [13, 26, 27, 28, 66]. One of the proposed approaches is the Subclass Discriminant Analysis, that relaxes the assumptions on the unimodal distribution of each class by representing it with several subclasses and formulating the criterion accordingly. This allows to obtain better performance on the real-world data, which does not generally follow unimodal distributions, and potentially obtain higher dimensionality, as the rank of the between-class scatter matrix becomes higher, as shown in the next chapters. The benefits of using SDA can be seen from Fig. 1.1, where the data of class 2 follows 2 disjoint distributions. As the means of two classes lie close to each other, LDA fails to find a subspace that would discriminate the classes well, while SDA succeeds, as can be seen from the figure.

However, the limitation of the high computational complexity remains valid, as the solution still requires an eigendecomposition of a large matrix. To overcome this limitation, multiple solutions have been proposed over the past years, including incremental learning solutions [34], approximate solutions [24], and speed-up solutions [9, 25, 29, 30, 59]. In this thesis, a speed-up extension for overcoming this limitation is proposed.

So far we have considered the methods that rely on one representation of a set of samples. Such methods are known as single-view methods. Inspired by the human perception of the world, that is not only based on one source of information but on a combination of the audio, visual, and tactile signals, etc., various multi-view learning methods have been proposed [10, 32, 63, 64, 67]. Such methods refer to the problems where multiple descriptions of the same set of samples are available. These descriptions can come from different domains, e.g., in the problem of the classification of a video based

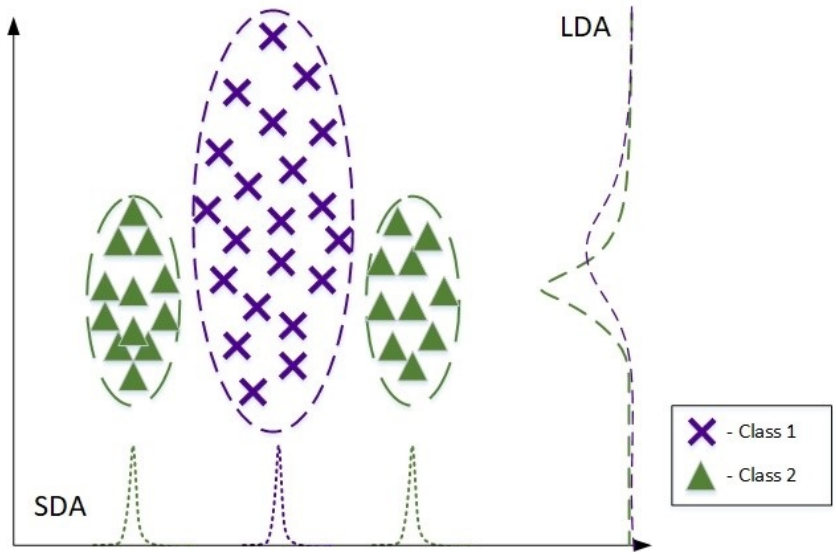

Figure 1.1. *Subclass data when projected to LDA subspace and SDA subspace*

on its audio and visual signals, or from different features of the same domain, e.g., in the problem of the classification of audio signal based on MFCC and ZCR. Different views can be also represented by the same type of features that are obtained from different data corresponding to the same sample, e.g., in the problem of person re-identification from multiple cameras.

Multiple extensions of LDA to multi-view problems have been proposed [10, 32, 65], but they mostly assume unimodality of classes in each view. In this thesis, an approach for overcoming this limitation by introducing a multi-view extension to Subclass Discriminant Analysis is proposed, and it is shown how the solution can be obtained in a fast and efficient way [15].

In order to address the issues of already existing methods as outlined earlier, we formulate the research problem of this work as the development of a method that would satisfy the following requirements:

- relax the assumptions of unimodality of each class
- be computationally efficient in both linear and non-linear formulations
- have the possible dimensionality higher than the number of classes -1
- be able to extend to multi-view data.

This thesis is organized as follows: in chapter 2, the previous work relevant to the further development of the proposed extensions is described, including the relevant subspace learning methods, kernel methods, and speed-up approaches. In chapter 3, the proposed extensions for Subclass Discriminant Analysis are described. In chapter 4, the experimental setup along with the datasets used for evaluation of the methods are described and the results are reported. In chapter 5, conclusions are made regarding the proposed approaches.

2 THEORETICAL BACKGROUND

2.1 Subspace Learning

In this section, the related work in the area of subspace learning is described. The goal of subspace learning methods is to find a subspace of lower dimensionality, projection onto which would result in the low-dimensional data while satisfying some criterion defined for the original data. Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N]$ be a set of N D-dimensional vectors in some space $\mathbb{R}^D$, and let each vector $\mathbf{x}_i$ correspond to a class label c_i. Using these notations, we can define the problem of dimensionality reduction as finding a feature space, defined by the projection matrix $\mathbf{W}$, such that its dimensionality is smaller than that of original space, and data belonging to different classes lies far from each other when projected onto it, while samples of the same class lie close to each other.

Prevailing part of the subspace learning methods find such a feature space by optimizing the Fisher-Rao's criterion [19, 46], that is defined over two symmetric positive semi-definite matrices, referred to as within-class scatter matrix $\mathbf{S}_w$ and between-class scatter matrix $\mathbf{S}_b$:

$$\mathcal{J}(\mathbf{W}) = \underset{\mathbf{W}}{argmin}\ \frac{Tr(\mathbf{W}^T\mathbf{S}_w\mathbf{W})}{Tr(\mathbf{W}^T\mathbf{S}_b\mathbf{W})}. \tag{2.1}$$

In other words, we want to minimize the within-class scatter of the data and maximize the between-class scatter while ensuring the orthogonality of the projection matrix.

The solution to the optimization problem in (2.1) is obtained by solving a generalized eigendecomposition problem [31]:

$$\mathbf{S}_w\mathbf{w} = \lambda\mathbf{S}_b\mathbf{w}, \tag{2.2}$$

and the projection matrix $\mathbf{W}$ is obtained by choosing the eigenvectors corresponding to minimal eigenvalues. The data in the feature space is obtained by a linear projection:

$$\mathbf{y}_i = \mathbf{W}^T\mathbf{x}_i. \tag{2.3}$$

Differences between various subspace learning methods primarily lie in the definitions of the between-class and the within-class scatter matrices. Further in this chapter, we will focus on different approaches taken in this area.

2.2 Linear and Nonlinear Learning

Being based on a linear projection, linear subspace learning approach described in the previous section assumes linear separability of data and cannot take into account possible nonlinearities that are often present in the real-world data. An example of such data is outlined in Fig. 2.1. To address this issue, multiple approaches have been taken, including kernel methods, neural network-based methods, random projections-based methods, etc. In this section, we focus on two of the commonly used approaches: a kernel-based approach that allows to learn the nonlinear decision boundaries without explicitly mapping the data to the nonlinear space, and the Nonlinear Projection Trick approach.

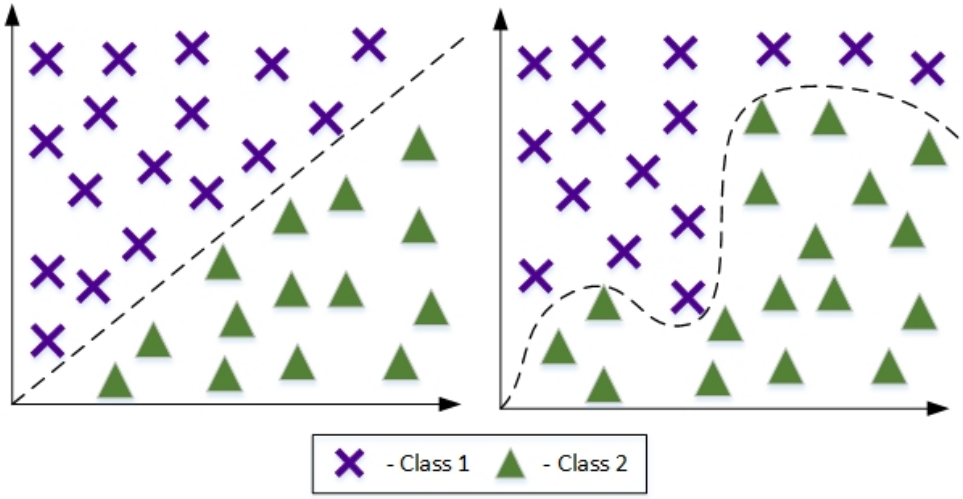

Figure 2.1. *Linear decision boundary vs. nonlinear decision boundary.*

The idea of the two approaches lies in the following: rather than finding a linear projection of the data in $\mathbf{X} \in \mathbb{R}^D$ directly, our goal is to first map it to some nonlinear feature space $\mathcal{F}$. The nonlinear feature space is defined by some nonlinear function $\phi()$, i.e., $\phi(\mathbf{x}_i) \in \mathcal{F}$. The dimensionality of $\mathcal{F}$ depends on the choice of the nonlinear function and can be infinite. A linear projection is then defined in $\mathcal{F}$, i.e. $\mathbf{y}_i = \mathbf{W}^T \phi(\mathbf{x}_i)$.

2.2.1 Kernel Trick

The explicit mapping of each sample in $\mathbf{X}$ to its image $\boldsymbol{\phi}_i = \phi(\mathbf{x}_i)$ raises issues related to the arbitrary dimensionality of $\mathcal{F}$, which can be infinite. To address this issue and avoid the explicit learning of nonlinear function $\phi()$ and mapping of $\mathbf{X}$ to $\boldsymbol{\Phi} = \phi(\mathbf{X})$ directly, kernel functions can be used. A kernel function $k()$ is defined over a pair of samples in $\mathbf{X}$ and maps their inner product in $\mathbb{R}^D$ to the corresponding inner product of their images in $\mathcal{F}$: $k(\mathbf{x}_1, \mathbf{x}_2) = \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_2)$. Thus, formulation of the subspace learning problem in a way that only relies on the inner products of data, rather than on the data directly, allows obtaining an efficient nonlinear solution.

The kernel matrix $\mathbf{K}$ of size $N \times N$ is, therefore, defined as $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. It is trivial to see that since $k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j), \mathbf{K} = \mathbf{\Phi}^T \mathbf{\Phi}$, where $\mathbf{\Phi} = [\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), ..., \phi(\mathbf{x}_N)]$. The kernel matrix $\mathbf{K}$ is generally assumed to be non-singular, while in the case of a singular matrix it is generally regularized as $\mathbf{K}^{'} = \mathbf{K} + \delta \mathbf{I}$, where δ is a regularization parameter. According to the Representer Theorem [49], $\mathbf{W}$ can be represented as a linear combination of data in $\mathcal{F}$:

$$\mathbf{W} = \mathbf{\Phi A}. \tag{2.4}$$

Hence, $\mathbf{y}_i = \mathbf{W}^T \phi(\mathbf{x}_i) = \mathbf{A}^T \mathbf{\Phi}^T \phi(\mathbf{x}_i) = \mathbf{A}^T \mathbf{k}_i$.

2.2.2 Nonlinear Projection Trick

Although kernel methods are widely used in machine learning problems where nonlinearity is required, there exist many optimization problems which are impossible to formulate solely with inner products, making the application of kernel trick impossible. For such problems, an approach referred to as Nonlinear Projection Trick (NPT) has been proposed [35]. In NPT, data samples are mapped to a nonlinear space of reduced dimensionality, referred to as effective space, directly.

Consider the eigendecomposition of a centered kernel matrix $\mathbf{K}$:

$$\mathbf{K} = \mathbf{U \Lambda U}^T, \tag{2.5}$$

where $\mathbf{U}$ is the matrix containing the eigenvectors of $\mathbf{K}$ corresponding to eigenvalues in $\mathbf{\Lambda}$, and $\mathbf{\Lambda}$ is a diagonal matrix of eigenvalues of $\mathbf{K}$ in decreasing order. The centering of $\mathbf{K}$ can be achieved as follows: [50]

$$\mathbf{K} = (\mathbf{I} - \mathbf{E}_N)\mathbf{K}(\mathbf{I} - \mathbf{E}_N), \tag{2.6}$$

$$\mathbf{E}_N = \frac{1}{N} \mathbf{1}_N \mathbf{1}_N^T, \tag{2.7}$$

where $\mathbf{1}_N \in \mathbb{R}^N$is a vector of ones. The feature representation $\mathbf{Y}$ in the effective subspace is then obtained as

$$\mathbf{Y} = \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{U}^T. \tag{2.8}$$

Subsequently, any vector from the space of original dimensionality can be projected to the effective space as

$$\mathbf{y} = \mathbf{\Lambda}^{-\frac{1}{2}} \mathbf{U}^T \mathbf{k}, \tag{2.9}$$

where $\mathbf{k}$ is the kernel representation corresponding to the vector $\mathbf{x}$. By selecting the first n eigenvectors of $\mathbf{K}$, $n < N$, feature representation of reduced dimensionality can be obtained. Utilization of NPT allows applying linear methods directly on the NPT-projected data without requiring a formulation that is based on the inner products.

2.3 Subspace Learning Methods

In this section, relevant subspace learning methods are discussed. First, we focus on the commonly used Linear Discriminant Analysis, followed by the Graph Embedding framework and its extensions, and several commonly-used subclass-based methods. Further, we consider multi-view problems and methods relevant to the area, followed by the discussion of some of the speed-up approaches.

2.3.1 Linear Discriminant Analysis

One of the most well-known supervised subspace learning methods is Linear Discriminant Analysis (LDA) [60] that seeks to find such a projection space, projection onto which would result in data of different classes lying far from each other, while samples of the same class being mapped close to each other. An example of the desired projection can be seen in Fig 2.2.

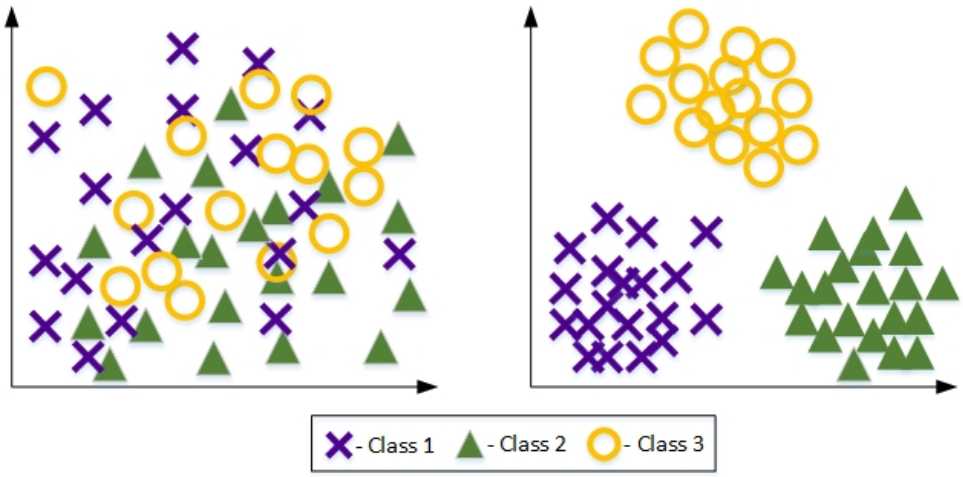

Figure 2.2. *Data before and after LDA projection.*

In LDA, each class is assumed to follow a unimodal Gaussian distribution and the objective of class separation is achieved by minimizing the Fisher-Rao's criterion (2.1), where the within-class scatter matrix is defined as

$$\mathbf{S}_w = \sum_{i=1}^{C} \sum_{j=1}^{N_i} (\mathbf{x}_{ij} - \boldsymbol{\mu}_i)(\mathbf{x}_{ij} - \boldsymbol{\mu}_i)^T, \tag{2.10}$$

and the between-class scatter matrix is defined as:

$$\mathbf{S_b} = \sum_{i=1}^{C} (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T, \tag{2.11}$$

where C is the number of classes, $\boldsymbol{\mu}$ is the mean of data, $\boldsymbol{\mu}_i$ is the mean of class i, N_i is the number of samples in class i and $\mathbf{x}_{ij}$ is the $j'th$ sample of class i.

Being a simplistic method, LDA has a number of limitations. The first limitation lies in the fact that the maximal dimensionality of the learned projection space is equal to the rank of the between-class scatter matrix, which is bounded by the number of classes, i.e., for the classification problem of C classes, maximal dimensionality of the learned subspace is equal to $C-1$. Another limitation of LDA lies in the assumption that each class follows a unimodal Gaussian distribution, which is generally not the case in the real-world problems, resulting in limited performance of the method.

2.3.2 Clustering-based Discriminant Analysis

One of the first approaches to overcoming the limitations of LDA on unimodality of classes was proposed in Clustering-based Discriminant Analysis (CDA) [13]. The method was first applied to a facial expression recognition problem, later extending to other subject areas. Inspired by the idea that facial expression data is not unimodal due to varying lighting or posture conditions, CDA proposes to represent the data of one class with several clusters. The clusters within each class are assumed to be known, as obtained by some clustering algorithm. CDA optimizes the following criterion:

$$\mathcal{J}(\mathbf{W}) = \underset{\mathbf{W}}{argmax}\ \frac{Tr(\mathbf{W}^T\hat{\mathbf{R}}\mathbf{W})}{Tr(\mathbf{W}^T\hat{\mathbf{C}}\mathbf{W})}, \tag{2.12}$$

$$\hat{\mathbf{R}} = \sum_{i=1}^{C-1}\sum_{l=i+1}^{C}\sum_{j=1}^{z_i}\sum_{h=1}^{z_l}(\boldsymbol{\mu}_j^{(i)} - \boldsymbol{\mu}_h^{(l)})(\boldsymbol{\mu}_j^{(i)} - \boldsymbol{\mu}_h^{(l)})^T, \tag{2.13}$$

$$\hat{\mathbf{C}} = \sum_{i=1}^{C}\sum_{j=1}^{z_i}\sum_{s}^{N_{i,j}}(\boldsymbol{x}_s - \boldsymbol{\mu}_j^{(i)})(\boldsymbol{x}_s - \boldsymbol{\mu}_j^{(i)})^T, \tag{2.14}$$

where $\mu_j^{(i)}$ represents the mean of class i and subclass j, z_i represents the number of subclasses in class i, $N_{i,j}$ is the number of elements in $j'th$ subclass of $i'th$ class, C is the total number of classes, and $\mathbf{x}_s$ represents the $s'th$ sample of subclass j in class i. The solution of (2.12) is obtained by solving the eigendecomposition problem

$$\hat{\mathbf{R}}\mathbf{w} = \lambda\hat{\mathbf{C}}\mathbf{w}, \tag{2.15}$$

and selecting the eigenvectors corresponding to maximal eigenvalues.

2.3.3 Kernel Clustering-based Discriminant Analysis

In order to address the possible nonlinearity of the problem, the kernelization of CDA (KCDA) was proposed [40]. The solution to KCDA is obtained by optimizing

$$\mathcal{J}(\mathbf{W}) = \underset{\mathbf{W}}{argmax}\ \frac{Tr(\mathbf{W}^T\mathbf{B}\mathbf{W})}{Tr(\mathbf{W}^T\mathbf{V}\mathbf{W})}, \tag{2.16}$$

$$\mathbf{B} = \sum_{i=1}^{C-1}\sum_{l=i+1}^{C}\sum_{j=1}^{z_i}\sum_{h=1}^{z_l}(\boldsymbol{\psi}^{i,j} - \boldsymbol{\psi}^{l,h})(\boldsymbol{\psi}^{i,j} - \boldsymbol{\psi}^{l,h})^T, \tag{2.17}$$

$$\mathbf{V} = \sum_{i=1}^{C}\sum_{j=1}^{z_i}\sum_{k=1}^{N_{i,j}}(\phi(\mathbf{x}_k^{i,j}) - \boldsymbol{\psi}^{i,j})(\phi(\mathbf{x}_k^{i,j}) - \boldsymbol{\psi}^{i,j})^T, \tag{2.18}$$

where C is the number of classes, z_i is the number of clusters in class i, $N_{i,j}$ is the number of elements in $j'th$ cluster of $i'th$ class. $\mathbf{x}_k^{i,j}$ is the $k'th$ sample of the $j'th$ cluster of the $i'th$ class, $\boldsymbol{\psi}^{i,j}$ is the center of the $j'th$ cluster of the $i'th$ class in the transformed space.

The solution to (2.16) is given by the eigendecomposition problem

$$\lambda\mathbf{V}\mathbf{v} = \mathbf{B}\mathbf{v}. \tag{2.19}$$

Let us assume that there exist such coefficients $\alpha_t^{r,s}$, $(r = 1, ..., C; s = 1, ..., z_r; t = 1, ..., N_{r,s})$, so that $\mathbf{v} = \sum_{r=1}^{C}\sum_{s=1}^{z_i}\sum_{t=1}^{N_{i,j}}\alpha_t^{r,s}\phi(\mathbf{x}_t^{r,s})$. Then the eigensystem in (2.19) is transformed to

$$\lambda\mathbf{S}_V\boldsymbol{\alpha} = \mathbf{S}_B\boldsymbol{\alpha}, \tag{2.20}$$

where the element of $\mathbf{S}_V$ corresponding to the column of $\alpha_t^{r,s}$ and the row of $\phi^T(\mathbf{x}_p^{m,n})$ is equal to

$$\mathbf{S}_V^{(\alpha_t^{r,s},\phi^T(\mathbf{x}_p^{m,n}))} = \sum_{i=1}^{C}\sum_{j=1}^{z_i}\sum_{k=1}^{N_{i,j}}\phi^T(\mathbf{x}_p^{m,n})(\phi(\mathbf{x}_k^{i,j} - \boldsymbol{\psi}^{i,j}))(\phi(\mathbf{x}_k^{i,j} - \boldsymbol{\psi}^{i,j}))^T\phi(\mathbf{x}_t^{r,s}). \tag{2.21}$$

An extensive derivation of (2.20) and (2.21) can be found in [40]. Similarly, the element of $\mathbf{S}_B$ corresponding to the column of $\alpha_t^{r,s}$ and the row of $\phi^T(\mathbf{x}_p^{m,n})$ is equal to

$$\mathbf{S}_B^{(\alpha_t^{r,s},\phi^T(\mathbf{x}_p^{m,n}))} = \sum_{i=1}^{C-1}\sum_{l=i+1}^{C}\sum_{j=1}^{z_i}\sum_{h=1}^{z_l}\phi^T(\mathbf{x}_p^{m,n})(\boldsymbol{\psi}^{i,j} - \boldsymbol{\psi}^{l,h}))(\boldsymbol{\psi}^{i,j} - \boldsymbol{\psi}^{l,h})^T\phi(\mathbf{x}_t^{r,s}), \tag{2.22}$$

$$\boldsymbol{\psi}^{i,j} = \frac{\sum_{k=1}^{N_{i,j}}\phi(\mathbf{x}_k^{i,j})}{N_{i,j}}. \tag{2.23}$$

By substituting (2.23) into (2.21) and (2.22), it is easy to see that only the inner product of the data points in the kernel space is required, so the kernel trick can be utilized.

2.3.4 Graph Embedding Framework

A general framework for unifying various dimensionality reduction methods has been proposed [52], where different subspace learning algorithms are considered from a graph embedding perspective. Data is described using the undirected weighted graph $\mathbf{G} = \{\mathbf{X}, \mathbf{W}\}$ with vertex set $\mathbf{X}$ and similarity matrix $\mathbf{W}$, where each vertex in $\mathbf{X}$ corresponds to a data sample. For each pair of vertices in $\mathbf{X}$, $\mathbf{W}$ measures their similarity by means of some similarity criterion, e.g., Gaussian similarity. Then, the diagonal degree matrix $\mathbf{D}$ and the Laplacian matrix $\mathbf{L}$ are defined as

$$\begin{aligned} \mathbf{L} &= \mathbf{D} - \mathbf{W}, \\ \mathbf{D}_{ii} &= \sum_{j \neq i} \mathbf{W}_{ij}, \end{aligned} \tag{2.24}$$

i.e., the degree matrix $\mathbf{D}$ at position (i, i) has the value of the sum of all values of $\mathbf{W}$ across $i'th$ row or column, as $\mathbf{W}$ is symmetric. The goal of graph embedding is therefore to find such a low-dimensional representation relationship among the vertices in $\mathbf{X}$ that incorporates the similarity relationship outlined in $\mathbf{G}$ in the best way.

The algorithm relies on two graphs: intrinsic graph and penalty graph. The intrinsic graph is the graph $\mathbf{G}$ itself, and the penalty graph, referred to as $\mathbf{G}^p = \{\mathbf{X}, \mathbf{W}^p\}$, represents the similarity characteristics of the data that are desired to be suppressed in the learned space. $\mathbf{X}$ is the same set of vertices as in $\mathbf{G}$, and $\mathbf{W}^p$ shows the similarity of vertex pairs from $\mathbf{X}$ that are to be suppressed.

Let us define a low-dimensional representation of vertices in $\mathbf{X}$ as $\mathbf{y} = [\mathbf{y}_1, ..., \mathbf{y}_i]$. Then, the objective function can be defined as follows:

$$\mathbf{y}^* = \underset{\mathbf{y}^T\mathbf{B}\mathbf{y}=d}{argmin} \sum_{i \neq j} ||\mathbf{y}_i - \mathbf{y}_j||^2 \mathbf{W}_{ij} = \underset{\mathbf{y}^T\mathbf{B}\mathbf{y}=d}{argmin} \mathbf{y}^T\mathbf{L}\mathbf{y}, \tag{2.25}$$

where d is a constant and $\mathbf{B}$ is the constraint matrix, typically $\mathbf{B} = \mathbf{L}^p = \mathbf{D}^p - \mathbf{W}^p$, where $\mathbf{D}^p$ is a diagonal degree matrix as defined in 2.24.

Assuming that $\mathbf{y} = \mathbf{X}^T\boldsymbol{\omega}$, i.e., low-dimensional representation of data can be obtained via linear projection, the objective function (2.25) can be reformulated as

$$\boldsymbol{\omega}^* = \underset{\substack{\boldsymbol{\omega}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\boldsymbol{\omega}=d \\ or\ \boldsymbol{\omega}^T\boldsymbol{\omega}=d}}{argmin} \sum_i ||\boldsymbol{\omega}^T\mathbf{x}_i - \boldsymbol{\omega}^T\mathbf{x}_j||^2 \mathbf{W}_{ij} = \underset{\substack{\boldsymbol{\omega}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\boldsymbol{\omega}=d \\ or\ \boldsymbol{\omega}^T\boldsymbol{\omega}=d}}{argmin} \boldsymbol{\omega}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\boldsymbol{\omega}, \tag{2.26}$$

where $\boldsymbol{\omega}$ is the projection vector.

The kernelized formulation of the method can be obtained similarly, assuming that $\boldsymbol{\omega} = \sum_i \alpha_i \phi(\mathbf{x}_i)$:

$$\mathbf{a}^* = \underset{\substack{\mathbf{a}^T\mathbf{KBK}^T\mathbf{a}=d \\ or\ \mathbf{a}^T\mathbf{Ka}=d}}{argmin} \sum_i ||\mathbf{a}^T\mathbf{K}_i - \mathbf{a}^T\mathbf{K}_j||^2\mathbf{W}_{ij} = \underset{\substack{\mathbf{a}^T\mathbf{KBK}^T\mathbf{a}=d \\ or\ \mathbf{a}^T\mathbf{Ka}=d}}{argmin} \mathbf{a}^T\mathbf{KLK}^T\mathbf{a}, \tag{2.27}$$

where $\mathbf{K}_i$ is the $i'th$ column of the kernel matrix $\mathbf{K}$.

Solutions of (2.25), (2.26) and (2.27) can be obtained by solving the generalized eigen-decomposition problem

$$\hat{\mathbf{L}}\mathbf{v} = \lambda\hat{\mathbf{B}}\mathbf{v}, \tag{2.28}$$

where $\hat{\mathbf{L}}$ is either $\mathbf{L}$, $\mathbf{XLX}^T$ or $\mathbf{KLK}$, and $\hat{\mathbf{B}}$ is $\mathbf{I}$, $\mathbf{B}$, $\mathbf{K}$, $\mathbf{XBX}^T$ or $\mathbf{KBK}$.

2.3.5 Marginal Fisher Analysis

Besides providing a new framework for representing various dimensionality reduction algorithms, the graph embedding framework can be used for the development of new algorithms, such as Marginal Fisher Analysis (MFA) [52]. In MFA, the intrinsic graph is designed to preserve the intra-class compactness and the penalty graph is designed to provide the inter-class separability. More specifically, intrinsic graph preserves the intra-class adjacency relationship by connecting each sample to it's k_{Int} closest neighbors within the class as defined by some similarity measure, such as Gaussian similarity based on the Euclidean distance. The penalty graph illustrates the marginal point adjacency relationship and connects the marginal point pairs of different classes. The example of such relationships can be seen in Fig. 2.3.

Thus, the intrinsic graph adjacency matrix $\mathbf{W}$ can be formed by setting $\mathbf{W}_{ij} = \mathbf{W}_{ji} = 1$ if x_i is among the k_{Int} nearest neighbors of x_j and they belong to the same class, and 0 otherwise. The penalty graph adjacency matrix $\mathbf{W}^p$ can be formed by setting $\mathbf{W}^p_{ij} = 1$ if the pair (i, j) is among the k_{Pen} shortest pairs among the set $\{(i, j), i \in \pi_c, j \notin \pi_c\}$. The Marginal Fisher Criterion can then be defined as

$$\boldsymbol{\omega}^* = \underset{\boldsymbol{\omega}}{argmin}\ \frac{\boldsymbol{\omega}^T\mathbf{X}(\mathbf{D} - \mathbf{W})\mathbf{X}^T\boldsymbol{\omega}}{\boldsymbol{\omega}^T\mathbf{X}(\mathbf{D}^p - \mathbf{W}^p)\mathbf{X}^T\boldsymbol{\omega}}, \tag{2.29}$$

which is a special case of the linearization of the graph embedding framework, where $\mathbf{B} = \mathbf{D}^p - \mathbf{W}^p$, and, therefore, can be solved accordingly.

The kernelization of MFA is obtained similarly by optimizing

$$\mathbf{a}^* = \underset{\mathbf{a}}{argmin}\ \frac{\mathbf{a}^T\mathbf{K}(\mathbf{D} - \mathbf{W})\mathbf{K}^T\mathbf{a}}{\mathbf{a}^T\mathbf{K}(\mathbf{D}^p - \mathbf{W}^p)\mathbf{K}^T\mathbf{a}}, \tag{2.30}$$

$$\boldsymbol{\omega} = \sum_{i=1}^{N} \mathbf{a}_i\phi(x_i). \tag{2.31}$$

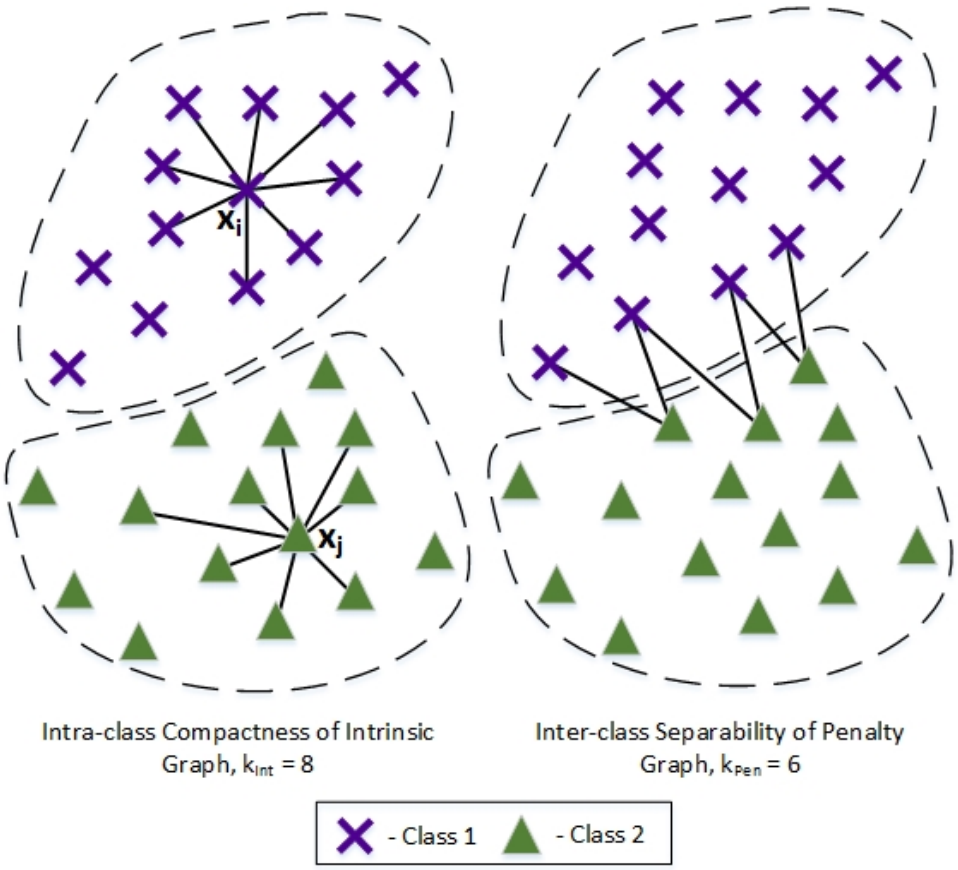

Figure 2.3. *Marginal Fisher Analysis graphs adjacency relationships.*

It should be noted that the projection into the feature space might result in different sets of nearest k_{Int} neighbors of the same class and k_{Pen} neighbors of a different class of each sample, resulting in different intrinsic and penalty graphs [52]. Based on the formulation of the Euclidean distance between two samples in the feature space

$$D(x_i, x_j) = \sqrt{k(x_i, x_i) + k(x_j, x_j) - 2k(x_i, x_j)}, \tag{2.32}$$

the optimal projection for a new data sample $\mathbf{x}$ is obtained as

$$\begin{aligned} F(\mathbf{x}, \mathbf{a}^*) &= \lambda \sum_{i=1}^{n} \mathbf{a}_i^* k(\mathbf{x}, \mathbf{x}_i), \\ \lambda &= \left(\mathbf{a}^{*T} \mathbf{K} \mathbf{a}^*\right)^{-\frac{1}{2}}. \end{aligned} \tag{2.33}$$

2.3.6 Subclass Graph Embedding Framework

Subclass Graph Embedding (SGE) framework has been proposed as an extension to the GE framework incorporating subclass information of the data [41]. Similarly to GE, the framework is based on the representation of data from a graph embedding perspective. SGE defines an affinity matrix $\mathbf{P}$ that is a block matrix with different blocks corresponding

to different subclasses, where $\mathbf{P}^{ij}(q,p)$ denotes the value at (q,p) position of the block corresponding to $i'th$ class and $j'th$ subclass. The first objective is the minimization of the within-class scatter and it is defined as:

$$\underset{\mathbf{V}}{argmin}(Tr(\mathbf{V}^T\mathbf{X}\mathbf{L}_{int}\mathbf{X}^T\mathbf{V})),\ \mathbf{L}_{int}=\mathbf{D}_{int}-\mathbf{W}_{int}, \tag{2.34}$$

$$\mathbf{W}_{int}=diag(\mathbf{W}_{int}^1,...,\mathbf{W}_{int}^c),\quad \mathbf{W}_{int}^i=diag(\mathbf{P}^{i1},...,\mathbf{P}^{id_i}). \tag{2.35}$$

The degree intrinsic matrix $\mathbf{D}_{int}$, defined similarly to GE framework, takes values:

$$\mathbf{D}_{int}(\sum_{s=0}^{i-1}\sum_{t=0}^{j-1}n_{st}+q,\sum_{s=0}^{i-1}\sum_{t=0}^{j-1}n_{st}+q)=\sum_{p}\mathbf{P}^{ij}(q,p), \tag{2.36}$$

where $diag()$ represents the block-diagonal matrix of corresponding blocks, $\mathbf{P}^{ij}$ is the block of $\mathbf{P}$ corresponding to the $j'th$ subclass of the $i'th$ class, and n_{st} is the number of samples in class s and subclass t, and d_i is the number of subclasses in class i.

Let us define the matrix $\mathbf{Q}$, where $\mathbf{Q}_{ij}^{lh}$ denotes the similarity between the mean vectors of different subclasses $\boldsymbol{\mu}^{ij}$ and $\boldsymbol{\mu}^{lh}$, where $\boldsymbol{\mu}^{ij}$ is the mean of the $j'th$ subclass of the $i'th$ class. The second objective is then the maximization of the between-class scatter and is defined as:

$$\underset{\mathbf{V}}{argmax}(Tr(\mathbf{V}^T\mathbf{X}\mathbf{L}_{pen}\mathbf{X}^T\mathbf{V})),\quad \mathbf{L}_{pen}=\mathbf{D}_{pen}-\mathbf{W}_{pen},\quad \mathbf{D}_{pen}=0 \tag{2.37}$$

$$\mathbf{W}_{pen}=\begin{pmatrix}\mathbf{W}_{pen}^{1,1} & \mathbf{W}_{pen}^{1,2} & ... & \mathbf{W}_{pen}^{1,c}\\ \mathbf{W}_{pen}^{2,1} & \mathbf{W}_{pen}^{2,2} & ... & \mathbf{W}_{pen}^{2,c}\\ ... & ... & ... & ...\\ \mathbf{W}_{pen}^{c,1} & \mathbf{W}_{pen}^{c,2} & ... & \mathbf{W}_{pen}^{c,c}\end{pmatrix}, \tag{2.38}$$

where the matrices on the main diagonal are given by

$$\mathbf{W}_{pen}^{i,i}=diag(\mathbf{W}^{i1},...,\mathbf{W}^{id_i}),\quad \mathbf{W}^{ij}=\frac{(\sum_{\omega\neq i}\sum_{t=1}^{d_\omega}\mathbf{Q}_{ij}^{\omega t})}{(n_{ij})^2}\mathbf{e}^{n_{ij}}\mathbf{e}^{n_{ij}T}, \tag{2.39}$$

where $\mathbf{e}^{n_{ij}}$is the vector of ones of length n_{ij}. The off-diagonal blocks are given as follows:

$$\mathbf{W}^{i,l}=\begin{pmatrix}\mathbf{W}_{i1}^{l1} & \mathbf{W}_{i1}^{l2} & ... & \mathbf{W}_{i1}^{ld_l}\\ \mathbf{W}_{i2}^{l1} & \mathbf{W}_{i2}^{l2} & ... & \mathbf{W}_{i2}^{ld_l}\\ ... & ... & ... & ...\\ \mathbf{W}_{id_i}^{l1} & \mathbf{W}_{id_i}^{l2} & ... & \mathbf{W}_{id_i}^{ld_l}\end{pmatrix}, i\neq l,\ \mathbf{W}_{ij}^{lh}=\frac{\mathbf{Q}_{ij}^{lh}}{n_{ij}n_{lh}}\mathbf{e}^{n_{ij}}\mathbf{e}^{n_{lh}T}. \tag{2.40}$$

The solution is then given by the eigendecomposition problem

$$\mathbf{X}\mathbf{L}_{int}\mathbf{X}^T\mathbf{v}=\lambda\mathbf{X}\mathbf{L}_{pen}\mathbf{X}^T. \tag{2.41}$$

SGE framework provides an extension to the Graph Embedding framework, allowing to reformulate the existing subclass-based methods, including CDA, LDA, PCA [16], and SDA that is discussed further. In addition, SGE allows developing new methods as outlined in the next section.

2.3.7 Subclass Marginal Fisher Analysis

To overcome limitations of Marginal Fisher Analysis related to the assumptions on the distribution of data, an extension incorporating subclass information has been proposed relying on SGE framework. This method is referred to as Subclass Marginal Fisher Analysis (SMFA) [42], and as MFA, it defines the intrinsic and penalty graph matrices relying on the nearest neighbor information of the graph vertices. The intrinsic graph matrix $\mathbf{P}^{ij}$ and the penalty graph matrix $\mathbf{W}_{pen}^{i,l}$ are defined as follows:

$$\mathbf{P}^{ij}(q,p) = \begin{cases} 1, \text{ if } p \in N_{k_{Int}}(q) \text{ or } q \in N_{k_{Int}}(p) \\ 0, \text{ otherwise} \end{cases}, \tag{2.42}$$

$$\mathbf{W}_{pen}^{i,l}(q,p) = \begin{cases} 1, \text{ if } i \neq l \text{ and } (p \in M_{k_{Pen}}(q) \text{ or } q \in M_{k_{Pen}}(p)) \\ 0, \text{otherwise} \end{cases}, \tag{2.43}$$

where $\mathbf{P}^{ij}(q,p)$ is the value at the (q,p) position of the $i'th$ class and $j'th$ subclass, $\mathbf{W}_{pen}^{i,l}(q,p)$ refers to the value at (q,p) position of the penalty matrix, where q belongs to the $i'th$ class, p belongs to the $l'th$ class, $N_{k_{Int}}(q)$ refers to the k_{Int} nearest neighbors of sample q, and $M_{k_{Pen}}$ refers to the k_{Pen} nearest neighbor samples of q outside class i. The nearest neighbors are selected based on the Gaussian similarity.

It can be noted that in SMFA the penalty graph matrix is designed in a way to ensure interclass separability, while intrinsic graph matrix ensures the intra-subclass compactness. We can also note that the penalty graph matrix does not exploit the subclass information, avoiding the constraints between subclasses of the same class.

SMFA offers another advantage compared to LDA and CDA - the maximal dimensionality of the projection space is defined by the parameter k_{Pen} hence offering much larger potential dimensionality than that of LDA or CDA. On the other hand, SMFA requires careful selection of the parameters of k_{Int} and k_{Pen} to ensure the generalizability of the method, while neither CDA nor LDA requires such an extensive parameter search.

2.3.8 Subclass Discriminant Analysis

Another approach to overcome the limitations caused by the assumptions on the unimodality of the data were introduced in Subclass Discriminant Analysis (SDA) [66]. Similarly to CDA and SMFA, the method uses a data representation where the data of the same class consists of several subclasses, that are assumed to be known and can be obtained by some clustering algorithm. The differences from CDA lie in the definitions of the between-class scatter matrix, where the prior probabilities of each subclass are added. Then, instead of minimizing the within-class scatter, SDA minimizes the total scatter of the matrix, which implicitly results in minimization of the within-class scatter, given that between-class scatter is maximized. The total scatter matrix and the between-class scatter matrix are defined as

$$\mathbf{S}_t = \sum_{q=1}^{N} (\mathbf{x}_q - \boldsymbol{\mu})(\mathbf{x}_q - \boldsymbol{\mu})^T, \tag{2.44}$$

$$\mathbf{S}_b = \sum_{i=1}^{C-1} \sum_{l=i+1}^{C} \sum_{j=1}^{d_i} \sum_{h=1}^{d_l} p_{ij} p_{lh} (\boldsymbol{\mu}_{ij} - \boldsymbol{\mu}_{lh})(\boldsymbol{\mu}_{ij} - \boldsymbol{\mu}_{lh})^T, \tag{2.45}$$

where i and l are the class labels, j and h are the subclass labels, N is the total number of samples in $\mathbf{X}$, $\boldsymbol{\mu}$ is the mean of data, $p_{ij} = \frac{N_{ij}}{N}$, where N_{ij} is the number of samples in subclass j of class i.

Instead of solving the minimization problem similar to (2.1), the objective of SDA can be reformulated into the maximization problem, and the definitions of the total scatter and between-class scatter can be reformulated relying on the graph embedding framework:

$$\mathcal{J}(\mathbf{W}) = \underset{\mathbf{W}}{argmax} \frac{Tr(\mathbf{W}^T \mathbf{S}_b \mathbf{W})}{Tr(\mathbf{W}^T \mathbf{S}_t \mathbf{W})}, \tag{2.46}$$

$$\mathbf{S}_b = \mathbf{X} \mathbf{L}_b \mathbf{X}^T, \tag{2.47}$$

$$\mathbf{L}_b(i,j) = \begin{cases} \frac{N - N_{c_i}}{N^2 N_{c_h}}, & \text{if } z_i = z_j = h \\ 0, & \text{if } z_i \neq z_j,\ c_i = c_j \\ -\frac{1}{N^2}, & \text{if } c_i \neq c_j \end{cases}, \tag{2.48}$$

where N_c is the number of samples in class c, c_i is the class label of $\mathbf{x}_i$, z_i is the subclass label of $\mathbf{x}_i$, and N_{ch} is the number of samples in subclass h of class c. It is trivial to see that for the mean-centered data definition of the total scatter becomes $\mathbf{S}_t = \mathbf{X}\mathbf{X}^T$. It can be seen that the solution to SDA is now obtained by solving an eigendecomposition problem

$$\mathbf{L}_b \mathbf{X}^T \mathbf{v} = \lambda \mathbf{X}^T \mathbf{v}. \tag{2.49}$$

2.3.9 Kernel Subclass Discriminant Analysis

The reformulations outlined in the previous section allow to simplify the formulation of the kernelized form of the algorithm and assist us in the development of further extensions [12]. The total scatter and between-class scatter of mean-centered data in $\mathcal{F}$ can be defined as

$$\mathbf{S}_{kt} = \sum_{i=1}^{N}(\boldsymbol{\phi}_i - \bar{\boldsymbol{\phi}})(\boldsymbol{\phi}_i - \bar{\boldsymbol{\phi}})^T = \boldsymbol{\Phi}\boldsymbol{\Phi}^T, \tag{2.50}$$

$$\begin{aligned}\mathbf{S}_{kb} = \sum_{i=1}^{C-1}\sum_{l=i+1}^{C}\sum_{j=1}^{d_i}\sum_{h=1}^{d_l} p_{ij}p_{lh}(\bar{\boldsymbol{\phi}}_{ij} - \bar{\boldsymbol{\phi}}_{lh})(\bar{\boldsymbol{\phi}}_{ij} - \bar{\boldsymbol{\phi}}_{lh})^T \\ = \boldsymbol{\Phi}\mathbf{L}_b\boldsymbol{\Phi}^T,\end{aligned} \tag{2.51}$$

where $\bar{\boldsymbol{\phi}}_{ij}$ is the mean of the subclass j of class i in $\mathcal{F}$, and $\bar{\boldsymbol{\phi}}$ is the mean of the data in $\mathcal{F}$. The solution to the kernelized formulation of SDA (KSDA) is then obtained by optimizing the objective function (2.46), which is solved by the generalized eigendecomposition problem

$$\boldsymbol{\Phi}\mathbf{L}_b\boldsymbol{\Phi}^T\boldsymbol{\Phi}\mathbf{a} = \lambda\boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{\Phi}\mathbf{a} => \tag{2.52}$$

$$\mathbf{K}\mathbf{L}_b\mathbf{K}\mathbf{a} = \lambda\mathbf{K}\mathbf{K}\mathbf{a} => \mathbf{L}_b\mathbf{K}\mathbf{a} = \lambda\mathbf{K}\mathbf{a}. \tag{2.53}$$

It can be noted that the solutions to the eigendecomposition problems (2.49) and (2.53) suffer from the limitations related to the high computational complexity in the cases of high-dimensional data and a large number of samples, respectively. In this work, a solution to overcoming this limitation is proposed.

2.4 Multi-view Learning

In some problems, description of the same instances of data can be available from different domains or feature types. Exploitation of such enriched information can potentially increase the performance of the machine learning algorithm. Different feature types can be represented by images, text, audio features, etc., and are referred to as views. It should be noted that multiple views do not necessarily come from multiple/different sources/sensors, as multiple different features can be extracted from the same source, e.g., MFCC features, pitch histogram, or RMS energy can describe the same audio file.

In the cases of multi-view data, one of the approaches is to concatenate the feature sets from different modalities and perform the subspace learning using traditional single-view methods. However, this can result in poor performance in the cases where the different modalities are represented by different distributions, or when some of the modalities are more useful than others. Therefore, a better approach to solving such problems lies in

finding a common projection space for all the modalities, projection onto which would result in the best separability of the data of different classes, as defined by some criterion. Such an approach is referred to as multi-view or multi-modal learning [55]. In addition, multi-view methods allow to address the situation where the data from one of the views becomes unavailable during testing, while utilization of concatenated features is impossible in such scenario.

2.4.1 Multi-view Extensions to Linear Discriminant Analysis

Let us consider data of V views $\mathbf{X} = diag(\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_V)$. We are looking for V projection matrices $\mathbf{W}_v$ of $d_v \times N$ dimensionality that project the data $\mathbf{X}_v$ from the views $v = [1, ..., V]$ to the latent space, such that a certain criterion is satisfied in that space. Generally, we want to keep distinct classes far from each other, while keeping the data compact.

In this section, we focus on several multi-view extensions to Linear Discriminant Analysis that are based on a generalized framework for multi-view subspace learning that has been recently proposed in [10]. Besides proposing extensions to LDA, this framework allows reformulating many of the existing methods.

Optimization criterion to be satisfied in the latent space is defined via the Fisher-Rao's criterion:

$$\mathcal{J}(\mathbf{W}) = \underset{\mathbf{W}}{argmax}\, \frac{Tr(\mathbf{W}^T\mathbf{P}\mathbf{W})}{Tr(\mathbf{W}^T\mathbf{Q}\mathbf{W})}, \tag{2.54}$$

which is solved by the generalized eigendecomposition problem

$$\mathbf{P}\mathbf{W} = \boldsymbol{\rho}\mathbf{Q}\mathbf{W}, \tag{2.55}$$

$$\mathbf{W} = \begin{pmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \\ ... \\ \mathbf{W}_V \end{pmatrix}, \tag{2.56}$$

where $\mathbf{W}_v$ is the projection matrix of the view v. $\mathbf{P}$ and $\mathbf{Q}$ are referred to as inter-view and intra-view covariance matrices. After finding $\mathbf{W}$, the feature vectors for the data of the $v'th$ view X_v in the latent space are obtained as $\mathbf{Y}_v = \mathbf{W}_v^T\mathbf{X}_v$. The inter-view and intra-view covariance matrices $\mathbf{P}$ and $\mathbf{Q}$ can be defined using the graph embedding framework as follows:

$$\mathbf{P} = \mathbf{X}\mathbf{L}_b\mathbf{X}^T, \tag{2.57}$$

$$\mathbf{Q} = \mathbf{X}\mathbf{L}_w\mathbf{X}^T, \tag{2.58}$$

$$\mathbf{X} = \begin{pmatrix} \mathbf{X}_1 & 0 & ... & 0 \\ 0 & \mathbf{X}_2 & ... & 0 \\ 0 & 0 & ... & \mathbf{X}_V \end{pmatrix}, \tag{2.59}$$

$$\mathbf{L}_b = \begin{pmatrix} \mathbf{L}_{b11} & \mathbf{L}_{b12} & ... & \mathbf{L}_{bV1} \\ \mathbf{L}_{b12} & \mathbf{L}_{b22} & ... & \mathbf{L}_{bV2} \\ ... & ... & ... & ... \\ \mathbf{L}_{b1V} & \mathbf{L}_{b2V} & ... & \mathbf{L}_{bVV} \end{pmatrix}, \tag{2.60}$$

$$\mathbf{L}_w = \begin{pmatrix} \mathbf{L}_{w11} & 0 & ... & 0 \\ 0 & \mathbf{L}_{w22} & ... & 0 \\ ... & ... & ... & ... \\ 0 & 0 & ... & \mathbf{L}_{wVV} \end{pmatrix}, \tag{2.61}$$

where $\mathbf{L}_{wii}$ is the within-class Laplacian matrix, $\mathbf{L}_{bij}$ is the between-class Laplacian matrix, i and j are the view labels, and V is the number of views. This way, by exploiting different definitions of the between-class and within-class Laplacian matrices, new methods can be formulated.

In this section, we focus on two extensions of Linear Discriminant Analysis formulated using this framework - Multi-view Modular Discriminant Analysis (MvMDA) and Standard Multi-view Discriminant Analysis (SMvDA), the differences in which lie in the definitions of the between-class Laplacian matrix. Multi-view Modular Discriminant Analysis aims at maximizing the distance between the means of different classes wiyhin different views, thus considering the samples of the specific view. The between-class Laplacian matrix is defined as:

$$\hat{\mathbf{L}}_{bij} = 2\sum_{p=1}^{C}\sum_{q=1}^{C}(\frac{1}{N_p^2}\mathbf{e}_p\mathbf{e}_p^T - \frac{1}{N_pN_q}\mathbf{e}_p\mathbf{e}_q^T). \tag{2.62}$$

SMvDA instead maximizes the distances between the classes from all views and defines the between-class Laplacian matrix as

$$\mathbf{L}^*_{bij} = \begin{cases} 2\sum_{p=1}^{C}\sum_{\substack{q=1 \\ q\neq p}}^{C}(\frac{V}{N_p^2}\mathbf{e}_p\mathbf{e}_p^T - \frac{1}{N_pN_q}\mathbf{e}_p\mathbf{e}_q^T), & \text{if } i = j \\ -2\sum_{p=1}^{C}\sum_{\substack{q=1 \\ q\neq p}}^{C}\frac{1}{N_pN_q}\mathbf{e}_p\mathbf{e}_q^T, & \text{if } i \neq j \end{cases}, \tag{2.63}$$

where i and j are views, C is the number of classes, and $\mathbf{e}_p$ is N -dimensional class vector with ones at the positions of instances of class p and zeros elsewhere.

Both of the SMvDA and MvMDA define the within-class Laplacian matrix similarly to single-view LDA:

$$\mathbf{L}_{wii} = \mathbf{I} - \sum_{c=1}^{C} \frac{1}{N_c} \mathbf{e}_c \mathbf{e}_c^T, \tag{2.64}$$

where C is the total number of classes, i is the view label, c is the class label, and $\mathbf{I}$ is the identity matrix.

The kernelized formulation of both methods can be obtained similarly by optimizing

$$\mathcal{J}(\mathbf{A}) = \underset{\mathbf{A}^T\mathbf{K}\mathbf{A}=\mathbf{I}}{argmax} \frac{Tr(\mathbf{A}^T\mathbf{P}^k\mathbf{A})}{Tr(\mathbf{A}^T\mathbf{Q}^k\mathbf{A})}, \tag{2.65}$$

$$\mathbf{P}^k = \mathbf{K}\mathbf{L}_b\mathbf{K}^T, \tag{2.66}$$

$$\mathbf{Q}^k = \mathbf{K}\mathbf{L}_w\mathbf{K}^T, \tag{2.67}$$

where $\mathbf{L}_b$ is defined using $\mathbf{L}^*_{bij}$ or $\hat{\mathbf{L}}_{bij}$ and $\mathbf{K}$ is a block-diagonal matrix having $\mathbf{K}_v$ as its $v'th$ block. This problem is solved by the eigendecomposition problem

$$\mathbf{P}^k\mathbf{A} = \boldsymbol{\rho}\mathbf{Q}^k\mathbf{A}. \tag{2.68}$$

A non-parametric version of MvMDA was also proposed in [11].

As can be seen from the formulations, being extensions of LDA, both of these methods suffer from the same limitation related to the assumption of unimodality of data of each class in each view. In this work, an approach for overcoming this limitation is proposed by introducing a multi-view extension to Subclass Discriminant Analysis that allows taking into account multiple distributions potentially present in data of one class in each view.

2.5 Speed-up Approaches

Most of the subspace learning methods rely on optimizing the Fisher-Rao's criterion that is solved by eigendecomposition of the scatter matrix or Laplacian matrix in the cases that rely on graph embeddings. Eigendecompositon of the scatter matrix suffers from high computational complexity in situations where the data is high-dimensional, and eigendecompositions of Laplacian matrix and of the scatter matrix in the kernelized forms are slow for large-scale datasets. In this section, we focus on several speed-up approaches taken in this area.

2.5.1 Spectral Regression Discriminant Analysis

An efficient solution to Linear Discriminant Analysis was proposed by introducing Spectral Regression [8]. Following the graph embedding framework, the between-class scatter of LDA can be formulated as

$$\mathbf{S}_b = \hat{\mathbf{X}}\mathbf{L}_b\hat{\mathbf{X}}^T, \tag{2.69}$$

$$\mathbf{L}_b = \begin{pmatrix} \mathbf{L}_b^{(1)} & 0 & ... & 0 \\ 0 & \mathbf{L}_b^{(2)} & ... & 0 \\ ... & ... & ... & ... \\ 0 & 0 & ... & \mathbf{L}_b^{(c)} \end{pmatrix}, \tag{2.70}$$

where $\hat{\mathbf{X}}$ is centered data matrix sorted according to class labels, $\mathbf{L}_b^{(k)}$ is $N_k \times N_k$ matrix with elements equal to $\frac{1}{N_k}$, and N_k is the number of elements in class k. The solution to LDA is then given by optimizing

$$\begin{aligned} \mathcal{J}(\mathbf{W}) &= \underset{\mathbf{W}}{argmax}\ \frac{Tr(\mathbf{W}^T\mathbf{S}_b\mathbf{W})}{Tr(\mathbf{W}^T\mathbf{S}_w\mathbf{W})}, \\ \text{equivalent to } &\underset{\mathbf{W}}{argmax}\ \frac{Tr(\mathbf{W}^T\mathbf{S}_b\mathbf{W})}{Tr(\mathbf{W}^T\mathbf{S}_t\mathbf{W})}, \end{aligned} \tag{2.71}$$

where $\mathbf{S}_t = \hat{\mathbf{X}}\hat{\mathbf{X}}^T$ for the mean-centered data. Exploiting (2.69), the solution to (2.71) is obtained by solving the eigendecomposition problem

$$\hat{\mathbf{X}}\mathbf{L}_b\hat{\mathbf{X}}^T\mathbf{w} = \lambda\hat{\mathbf{X}}\hat{\mathbf{X}}^T\mathbf{w}, \tag{2.72}$$

which is the main computational bottleneck of LDA.

By setting $\hat{\mathbf{X}}^T\mathbf{w} = \mathbf{t}$, the solution to (2.72) is obtained by solving the eigendecomposition problem $\mathbf{L}_b\mathbf{t} = \lambda\mathbf{t}$:

$$\hat{\mathbf{X}}\mathbf{L}_b\hat{\mathbf{X}}^T\mathbf{w} = \hat{\mathbf{X}}\mathbf{L}_b\mathbf{t} = \hat{\mathbf{X}}\lambda\mathbf{t} = \lambda\hat{\mathbf{X}}\mathbf{t} = \lambda\hat{\mathbf{X}}\hat{\mathbf{X}}^T\mathbf{w}. \tag{2.73}$$

Therefore, instead of solving the eigendecomposition problem in (2.72), we can solve the eigendecomposition problem $\mathbf{L}_b\mathbf{t} = \lambda\mathbf{t}$ and find such $\mathbf{w}$ so that $\hat{\mathbf{X}}^T\mathbf{w} = \mathbf{t}$. In reality, such $\mathbf{w}$ might not always exist, but can be approximated using least squares regression. Generally, regularized least squares regression is used, as for high-dimensional data, $\hat{\mathbf{X}}^T\mathbf{w} = \mathbf{t}$ is underdetermined:

$$\mathbf{W} = \underset{\mathbf{W}}{argmin}\ ||\mathbf{W}^T\mathbf{X} - \mathbf{T}||_2^2, \tag{2.74}$$

$$(\mathbf{X}\mathbf{X}^T + \alpha\mathbf{I})\mathbf{W} = \mathbf{X}\mathbf{T}^T, \tag{2.75}$$

$$\mathbf{W} = (\mathbf{X}\mathbf{X}^T + \alpha\mathbf{I})^{-1}\mathbf{X}\mathbf{T}^T, \tag{2.76}$$

where α is a regularization parameter and $\mathbf{T} = [\mathbf{t}_1, ..., \mathbf{t}_d]^T$.

Such formulation is beneficial, as the eigendecomposition of $\mathbf{L}_b$ can be solved trivially by following a fast process. It can be observed from (2.70) that the matrix $\mathbf{L}_b$ is block-diagonal, hence, its eigenpairs are the union of the eigenpairs of its blocks. In addition, $\mathbf{L}_b^{(k)}$ has a vector of ones as its eigenvector that corresponds to the eigenvalue zero, and the rank of $\mathbf{L}_b^{(k)}$ is one. Therefore, $\mathbf{L}_b$ has as many eigenvectors corresponding to non-zerp eigenvalues as there are blocks in the matrix, i.e., C. These eigenvectors correspond to the eigenvalue of one and follow a specific structure [7]:

$$\mathbf{u}_i = [\underbrace{0, ..., 0}_{\sum_{i=1}^{p-1} N_i}, \underbrace{1, ..., 1}_{N_p}, \underbrace{0, ..., 0}_{\sum_{i=p+1}^{C} N_i}]^T, \tag{2.77}$$

where p is the class label, N_p is the number of samples in class p and C is the number of classes.

As 1 is the eigenvalue of $\mathbf{L}_b$ that is repeated for the eigenvectors $\mathbf{u}_i$, any C linearly independent eigenvectors in the space spanned by $\mathbf{u}_i$ can be selected as eigenvectors of $\mathbf{L}_b$. At the same time, we can notice that the vector of ones is the eigenvector of $\mathbf{L}_b$, while it is orthogonal to $\hat{\mathbf{X}}$ [8]. Therefore, the vector of ones can be selected as the first eigenvector, and the rest eigenvectors from $\mathbf{u}_i$ can be orthogonalized from it. The vector of ones can then be removed.

By following this process, a solution to LDA is obtained with a linear-time complexity with respect to N or d, while the eigendecomposition-based approach has the cubic-time complexity in relation to $min(d, N)$, where N is the number of samples, and d is the dimensionality of the data.

2.5.2 Kernel Regression

Kernelized formulation of the Spectral Regression approach was proposed in [9]. In the kernel formulation of LDA, an objective can be defined based on the graph embedding framework similarly to the linear case described in the previous section. The solution is therefore given by the eigendecomposition problem $\mathbf{K}\mathbf{L}_b\mathbf{K}\mathbf{a} = \lambda\mathbf{K}\mathbf{K}\mathbf{a}$, which is equivalent to solving the eigendecomposition problem of $\mathbf{L}_b\mathbf{t} = \lambda\mathbf{t}$ given $\mathbf{K}\mathbf{a} = \mathbf{t}$:

$$\mathbf{K}\mathbf{L}_b\mathbf{K}\mathbf{a} = \mathbf{K}\mathbf{L}_b\mathbf{t} = \lambda\mathbf{K}\mathbf{t} = \lambda\mathbf{K}\mathbf{K}\mathbf{a}. \tag{2.78}$$

Kernel regression is then applied to obtain $\mathbf{a}$:

$$\mathbf{W}^* = \underset{\mathbf{W}}{argmin}\, ||\mathbf{W}^T\mathbf{\Phi} - \mathbf{T}||_2^2, \tag{2.79}$$

$$\mathbf{A} = \underset{\mathbf{A}}{argmin}\, ||\mathbf{A}^T\mathbf{\Phi}^T\mathbf{\Phi} - \mathbf{T}||_2^2 = \underset{\mathbf{A}}{argmin}\, ||\mathbf{A}^T\mathbf{K} - \mathbf{T}||_2^2, \tag{2.80}$$

$$\mathbf{A} = (\mathbf{K}\mathbf{K}^T + \alpha\mathbf{I})^{-1}\mathbf{K}\mathbf{T}^T, \tag{2.81}$$

where α is the regularization parameter. Here we can note that $\mathbf{L}_b$ has exactly the same structure as in the linear case, hence the eigendecomposition step in (2.78) can be solved by following exactly the same fast process as the one described for the linear case.

2.5.3 Approximate Kernel Regression

An approach for accelerating of kernel regression further has been proposed in [24, 30]. The idea lies in the substitution of the kernel matrix with an approximate kernel matrix, constructed from the inner products of data in the feature space with a set of r reference vectors $\mathbf{\Psi}$ in $\mathcal{F}$, $(r < N)$.

Reference vectors $\mathbf{\Psi}$ in $\mathcal{F}$ correspond to the prototype vectors in $\mathbb{R}^D$, which can be selected randomly from the training set; created randomly from the same distribution as the training data; obtained from clustering all data and selecting the centroids of the clusters; in the cases of subclass-based methods, i.e., SDA, CDA, or SMFA - centers of the subclasses.

Following this approach, $\mathbf{W}$ is represented by a linear combination of reference vectors $\mathbf{\Psi}$ as $\mathbf{W} = \mathbf{\Psi}\mathbf{A}$. Then, (2.80) becomes

$$\mathbf{A}^* = \underset{\mathbf{A}}{argmin}\, ||\mathbf{A}^T\mathbf{\Psi}^T\mathbf{\Phi} - \mathbf{T}||_2^2 = \underset{\mathbf{A}}{argmin}\, ||\mathbf{A}^T\hat{\mathbf{K}} - \mathbf{T}||_2^2, \tag{2.82}$$

where $\hat{\mathbf{K}} = \mathbf{\Psi}\mathbf{\Phi}$.

Then,

$$\mathbf{A} = (\hat{\mathbf{K}}\hat{\mathbf{K}}^T + \alpha\mathbf{I})^{-1}\hat{\mathbf{K}}\mathbf{T}^T, \tag{2.83}$$

where α is a regularization parameter. It should be noted that in the case $\mathbf{\Psi} = \mathbf{\Phi}$, the problem becomes equivalent to (2.81).

2.5.4 Nyström-based Approximate Kernel Subspace Learning

An approach to efficient subspace learning in the nonlinear feature space based on Nyström-based approximation has been proposed in [23]. Revisiting the Nonlinear Projection Trick, we can notice that in order to obtain the feature space of reduced dimensionality $n < N$, eigendecomposition of the kernel matrix $\mathbf{K}$ can be applied, keeping n leading eigenvalues and corresponding eigenvectors.

The optimal approximation of $\mathbf{K}$ is then given as

$$\mathbf{K} \approx \hat{\mathbf{Y}}^T\hat{\mathbf{Y}} = (\mathbf{\Lambda}_{(n)}^{\frac{1}{2}}\mathbf{U}_{(n)}^T)^T(\mathbf{\Lambda}_{(n)}^{\frac{1}{2}}\mathbf{U}_{(n)}^T), \tag{2.84}$$

where $\mathbf{\Lambda}_{(n)}$ is the diagonal matrix of n largest eigenvalues of $\mathbf{K}$, $\mathbf{U}_{(n)}$ is the matrix of eigenvectors corresponding to the eigenvalues in $\mathbf{\Lambda}_{(n)}$, and $\mathbf{Y}_{(n)}$ is the data in the effective n-dimensional space.

We can note that this solution requires still the construction and eigendecomposition of $\mathbf{K}$. As an approach to overcome this limitation, the Nyström method suggests the utilization of a subset of training samples to calculate the approximate kernel matrix. Let $\mathbf{K}_{Nn}$ be the $\mathcal{R}^{N\times n}$ matrix corresponding to the dot products of training data with n reference vectors, and $\mathbf{K}_{nn}$ - the kernel matrix of dot products of reference vectors with each other. Then, the kernel matrix $\mathbf{K}$ can be approximated as

$$\mathbf{K} \approx \mathbf{K}_{Nn}\mathbf{K}_{nn}^{-1}\mathbf{K}_{Nn}^T. \tag{2.85}$$

Although by following this approach the computational burden of the full kernel matrix calculation is omited, the eigendecomposition cost is still substantial. An approach to overcoming this limitation was proposed in [23]. We can note that the kernel matrix $\mathbf{K}$ can be approximated as follows:

$$\mathbf{K} \approx \mathbf{K}_{Nn}\mathbf{K}_{nn}^{-1}\mathbf{K}_{Nn}^T = (\mathbf{K}_{nn}^{-\frac{1}{2}}\mathbf{K}_{Nn}^T)^T(\mathbf{K}_{nn}^{-\frac{1}{2}}\mathbf{K}_{Nn}^T) = \hat{\mathbf{Y}}^T\hat{\mathbf{Y}}, \tag{2.86}$$

where $\mathbf{Y}_{(n)}$ is the data in the effective n-dimensional space. It can be observed that when the $\mathbf{K}_{nn}$ is an n-rank matrix, the matrices $\hat{\mathbf{Y}}\hat{\mathbf{Y}}^T$ and $\hat{\mathbf{Y}}^T\hat{\mathbf{Y}}$ are symmetrizable matrix products, hence, they have the same leading n eigenvalues. Since $\hat{\mathbf{Y}}\hat{\mathbf{Y}}^T$ is of $n \times n$ dimensions, applying eigendecomposition to it allows obtaining $\mathbf{\Lambda}_{(n)}$ with less computational cost than by approximating $\mathbf{K}$ as $\hat{\mathbf{Y}}^T\hat{\mathbf{Y}}$ and applying eigendecomposition to it. After obtaining the eigenvectors, the n-dimensional vector can be obtained by

$$\hat{\mathbf{y}} = \mathbf{\Lambda}_{(n)}^{-\frac{1}{2}}\mathbf{\Lambda}_n^{-1}\mathbf{U}_n^T\mathbf{K}_{Nn}\mathbf{k} = \mathbf{W}^T\phi(\mathbf{x}), \tag{2.87}$$

and the projection matrix can then be obtained as

$$\mathbf{W} = \mathbf{\Phi}\mathbf{K}_{Nn}\mathbf{U}_n\mathbf{\Lambda}_n^{-1}\mathbf{\Lambda}_{(n)}^{-\frac{1}{2}}. \tag{2.88}$$

Following this approach allows obtaining the subspace determined by n maximal eigenvalues of the optimal approximation of $\mathbf{K}$, while substantially reducing both the memory and time complexities.

2.5.5 Incremental Learning

In real-world problems, the situation is often such that the dataset is not available at once at the beginning of training, but becomes available in chunks. A naive approach to solving such problems would be to retrain the model every time new data is available, resulting in significant training time requirements. Other approaches propose to update the previously trained model by incorporating newly obtained data. Such approaches are referred to as incremental learning. As an example, let us consider an incremental solution for LDA [44].

Sequential Incremental LDA

LDA model can be represented by a discriminant eigenspace $\mathbf{\Omega} = (\mathbf{S_w}, \mathbf{S_b}, \boldsymbol{\mu}, N)$, where $\mathbf{S_w}$ and $\mathbf{S_b}$ are the within-class and between-class scatter matrix as defined in 2.10 and 2.11, $\boldsymbol{\mu}$ is the mean vector and N is the number of samples. Let M be the number of classes. Suppose that the new $(N+1)^{th}$ sample $\mathbf{y}$ corresponding to a class label k is added to the dataset. The problem can then be formulated as finding an updated discriminant eigenspace $\mathbf{\Omega}^{'} = (\mathbf{S_w}^{'}, \mathbf{S_b}^{'}, \boldsymbol{\mu}^{'}, N+1)$. The mean of the new data can be obtained as follows:

$$\boldsymbol{\mu}^{'} = \frac{N\boldsymbol{\mu} + \mathbf{y}}{N+1}, \tag{2.89}$$

In the case, where the class label k of a new sample $\mathbf{y}$ represents a new class, i.e., $k = M+1$, the between-class scatter matrix is defined as follows:

$$\begin{aligned} \mathbf{S_b}^{'} &= \sum_{c=1}^{M} n_c(\boldsymbol{\mu}_c - \boldsymbol{\mu}^{'})(\boldsymbol{\mu}_c - \boldsymbol{\mu}^{'})^T + (\mathbf{y} - \boldsymbol{\mu}^{'})(\mathbf{y} - \boldsymbol{\mu}^{'})^T \\ &= \sum_{c=1}^{M+1} n_c^{'}(\boldsymbol{\mu}_c - \boldsymbol{\mu}^{'})(\boldsymbol{\mu}_c - \boldsymbol{\mu}^{'})^T, \end{aligned} \tag{2.90}$$

where n_c is the number of samples in class c prior to the addition of a new sample, $n_c^{'}$ is a number of samples of class c after addition of a new sample, $n_c^{'} = n_c$ if $1 \leq c \leq M$, $n_c^{'} = 1$ if $c = M+1$, $\boldsymbol{\mu}_c = \mathbf{y}$ if $c = M+1$. The within-class scatter matrix does not change in this case:

$$\mathbf{S_w}^{'} = \sum_{c=1}^{M} \boldsymbol{\Sigma_c} + \boldsymbol{\Sigma_k} = \sum_{c=1}^{M} \boldsymbol{\Sigma}_c = \mathbf{S_w}. \tag{2.91}$$

In the case when y belongs to some already presented class, i.e., $k < M$:

$$\mathbf{S_b}^{'} = \sum_{c=1}^{M} n_c^{'}(\boldsymbol{\mu}_c^{'} - \boldsymbol{\mu}^{'})(\boldsymbol{\mu}_c^{'} - \boldsymbol{\mu}^{'})^T, \tag{2.92}$$

$$\boldsymbol{\mu}_k^{'} = \frac{n_k \boldsymbol{\mu}_k + \mathbf{y}}{n_k + 1}, \tag{2.93}$$

$$\mathbf{S_w}^{'} = \sum_{c=1, c \neq k}^{M} \boldsymbol{\Sigma}_c + \boldsymbol{\Sigma}_k^{'}, \tag{2.94}$$

where $\boldsymbol{\Sigma}_{\boldsymbol{k}}^{'}$ is an updated covariance matrix of class k,

$$\boldsymbol{\Sigma}_k^{'} = \boldsymbol{\Sigma}_k + \frac{n_k}{n_k + 1}(\mathbf{y} - \boldsymbol{\mu}_k^{'})(\mathbf{y} - \boldsymbol{\mu}_k^{'})^T, \tag{2.95}$$

where $\boldsymbol{\Sigma}_k$ is the covariance matrix of class k before the addition of a new sample, n_k is the number of samples in class k, and $\boldsymbol{\mu}_k^{'}$ is an updated mean of class k. After obtaining the updated within-class scatter matrix, between-class scatter matrix, the projection matrix can be obtained by following the procedure of LDA.

Chunk Incremental LDA

In addition to the Sequential Incremental LDA, where the samples are added one by one, the Chunk Incremental LDA has been proposed [44], providing a solution to the problem, where multiple new samples are added to the training data simultaneously. Let $\mathbf{Y}_t$ be a chunk of data samples at a time point t. Consider each chunk to be represented by a set of data samples $\{\mathbf{y}_1, ..., \mathbf{y}_L\}$, where L is the number of data samples in a chunk, $L \geq 1$. The problem can be formulated as finding an updated eigenspace Φ = ($\mathbf{S_w}^{'}, \mathbf{S_b}^{'}, \boldsymbol{\mu}^{'}, N + L$), similarly to the sequential case.

Considering the case where the samples of a new chunk belong to already existing classes, let l_c be the number of new samples in class c, and M be the total number of classes. Then, $n_c^{'} = n_c + l_c$, $N + L = \sum_{c=1}^{M} n_c^{'} = \sum_{c=1}^{M}(n_c + l_c)$. Hence,

$$\boldsymbol{\mu}_c^{'} = \frac{n_c \boldsymbol{\mu}_c + l_c \boldsymbol{\mu}_{yc}}{n_c + l_c}, \tag{2.96}$$

$$\boldsymbol{\mu}^{'} = \frac{N\boldsymbol{\mu} + L\boldsymbol{\mu}_y}{N + L}, \tag{2.97}$$

where $\boldsymbol{\mu}_{yc}$ is the mean of class c in the new chunk, $\boldsymbol{\mu}_c$ is the mean of class c prior to addition of a new chunk, and $\boldsymbol{\mu}_y = \frac{\sum_{j=1}^{L} \mathbf{y}_j}{L}$. The updated between-class scatter matrix, hence, can be defined as:

$$\mathbf{S_b}^{'} = \sum_{c=1}^{M} n_c^{'}(\boldsymbol{\mu}_c^{'} - \boldsymbol{\mu}^{'})(\boldsymbol{\mu}_c^{'} - \boldsymbol{\mu}^{'})^T. \tag{2.98}$$

The updated within-class scatter matrix becomes [44]:

$$\mathbf{S_w}^{'} = \sum_{c=1}^{M} \mathbf{\Sigma}_c^{'}, \tag{2.99}$$

$$\mathbf{\Sigma}_c^{'} = \mathbf{\Sigma}_c + \frac{n_c l_c^2}{n_c + l_c}(\mathbf{D_c}) + \frac{n_c^2}{(n_c + l_c)^2}(\mathbf{E}_c) + \frac{l_c(l_c + 2n_c)}{(n_c + l_c)^2}(\mathbf{F}_c), \tag{2.100}$$

$$\mathbf{D}_c = (\boldsymbol{\mu}_{yc} - \boldsymbol{\mu}_c)(\boldsymbol{\mu}_{yc} - \boldsymbol{\mu}_c)^T, \tag{2.101}$$

$$\mathbf{E}_c = \sum_{j=1}^{l_c} (\mathbf{y}_{cj} - \boldsymbol{\mu}_c)(\mathbf{y}_{cj} - \boldsymbol{\mu}_c)^T, \tag{2.102}$$

$$\mathbf{F}_c = \sum_{j=1}^{l_c} (\mathbf{y}_{cj} - \boldsymbol{\mu}_{yc})(\mathbf{y}_{cj} - \boldsymbol{\mu}_{yc})^T, \tag{2.103}$$

where $\mathbf{y}_{cj}$ is the $j'th$ sample of class c in the new chunk.

Taking into account the possibility of a new class being introduced in a new data chunk, without loss of generality, we can assume that a new chunk consists of l_{M+1} new samples belonging to class $M + 1$. Thus, the updated between-class scatter matrix can be formulated as:

$$\begin{aligned} \mathbf{S_b}^{'} &= \sum_{c=1}^{M} n_c^{'}(\boldsymbol{\mu}_c - \boldsymbol{\mu}^{'})(\boldsymbol{\mu}_c - \boldsymbol{\mu}^{'})^T + l_{M+1}(\boldsymbol{\mu}_y - \boldsymbol{\mu}^{'})(\boldsymbol{\mu}_y - \boldsymbol{\mu}^{'})^T \\ &= \sum_{c=1}^{M+1} n_c^{'}(\boldsymbol{\mu}_c - \boldsymbol{\mu}^{'})(\boldsymbol{\mu}_c - \boldsymbol{\mu}^{'})^T, \end{aligned} \tag{2.104}$$

where $n_c^{'}$ is the updated amount of elements in class c. In this case, the within-class scatter matrix can be reformulated as:

$$\mathbf{S_w}^{'} = \sum_{c=1}^{M} \mathbf{\Sigma}_c + \mathbf{\Sigma}_{M+1} = \sum_{c=1}^{M+1} \mathbf{\Sigma}_c^{'}, \tag{2.105}$$

$$\mathbf{\Sigma}_{M+1} = \sum_{i=1}^{l_{M+1}} (\mathbf{y}_i - \boldsymbol{\mu}_{y_{M+1}})(\mathbf{y}_i - \boldsymbol{\mu}_{y_{M+1}})^T, \tag{2.106}$$

where $\boldsymbol{\mu}_{y_{M+1}}$ is the mean of the new class in the new chunk.

Another approach to solving the problems of online learning through incremental subspace learning relies on incremental regression, and various solutions were proposed for different subspace learning methods [25].

3 PROPOSED METHODS

Having reviewed the related works in the area, we can now formulate several extensions. Firstly, we will formulate a solution to SDA via Spectral Regression. From this formulation, we will show how the solution can be obtained following a much faster process, relying on the structure of the between-class Laplacian matrix. Secondly, we will propose a novel criterion for extending the SDA into multi-view problems. We will further show how the solution to this criterion can be obtained by following a fast process similar to the one described for single-view SDA.

3.1 Spectral Regression Subclass Discriminant Analysis

In the previous chapter, we have considered how Spectral Regression can be used to obtain a solution for Linear Discriminant Analysis in a fast manner. Previously, Spectral Regression has not been exploited to solve Subclass Discriminant Analysis criterion, although this solution is straightforward and can be described as follows:

1. Create a between-class Laplacian scatter matrix according to (2.48)
2. Solve the eigendecomposition problem and create the matrix $\mathbf{T}$ out of the obtained vectors
$$\mathbf{L}_b\mathbf{t} = \lambda\mathbf{t} \tag{3.1}$$
3. Regress $\mathbf{T}$ to $\mathbf{W}$ following (2.76)
4. Orthogonalize $\mathbf{W}$ such that $\mathbf{W}^T\mathbf{W} = \mathbf{I}$

Equivalently, for the kernel case, the method can be formulated as follows:

1. Create a between-class Laplacian scatter matrix according to (2.48)
2. Solve the eigendecomposition problem (3.1) and create the matrix $\mathbf{T}$ out of the obtained vectors
3. Regress $\mathbf{T}$ to $\mathbf{A}$ according to (2.81) or (2.83)
4. Orthogonalize $\mathbf{A}$ such that $\mathbf{A}^T\mathbf{K}\mathbf{A} = \mathbf{I}$

It should be noted that the target vectors $\mathbf{t}$ are the same for the linear and non-linear cases. The exploitation of such formulation is beneficial as it allows to solve the resulting eigendecomposition problem by following a fast and straightforward process, as we will see further.

3.2 Speeding Up the Eigendecomposition Step

As mentioned earlier, subspace learning methods suffer from high computational complexity when the dimensionality of data is high, and in the kernelized methods, when the number of instances is high. In this section, a speed-up approach for Subclass Discriminant Analysis that allows overcoming this limitation and applying the method on large-scale high-dimensional data is presented. The proposed approach relies on substitution of the computationally intensive eigendecomposition step in (2.76) by a straightforward process that is dictated by the specific structure of between-class Laplacian matrix. Here and further, we assume that the data is mean-centered and the subclass label of each instance is known. We also assume that the data is sorted according to these labels from the first subclass of the first class to the last subclass of the last class.

Let us consider a binary classification problem with both classes containing 2 subclasses. Let the first subclass of the first class contain 3 samples, and the second subclass 5 samples. Let the first subclass of the second class contain 4 samples and the second 5 samples. The structure of $\mathbf{L}_b$ can be observed from Tab. 3.1 and the structure of corresponding eigenvectors from (3.2), where c corresponds to the class label, z to subclass label and r_i to $i'th$ random value.

We can see that the matrix consists of blocks, where each diagonal block corresponds to instances of a certain class. In Tab. 3.1, different blocks are highlighted with different colors for the sake of clarity, and v_1, v_2, v_3, and v_4 are positive constant values corresponding to the instances within the first and second subclass of the first and second class, respectively. v_5 is a constant negative value, showing the relations between instances of different classes.

Within each of the class blocks, a block structure showing the subclass structure of a class can be seen: the blocks corresponding to the instances of the same subclass have positive constant values, while the values corresponding to instances of different subclasses of the same class are zeros. The values corresponding to different classes are negative constant values. For data of C classes and Z subclasses in each class, the rank of $\mathbf{L}_b$ is equal to $C * Z - 1$, the matrix has the same amount of nonzero eigenvalues.

The specific block structure of $\mathbf{L}_b$ dictates the specific structure of its eigenvectors: $C - 1$ eigenvectors corresponding to the largest eigenvalues show the structure, from which the class label of each instance can be inferred, and the values at positions of the same class share the same value, and the values between classes are different. The rest of the eigenvectors show the subclass structure, where each eigenvector is responsible

Table 3.1. *Structure of the between-class Laplacian matrix in SDA.*

v_1	v_1	v_1	0	0	0	0	0	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5
v_1	v_1	v_1	0	0	0	0	0	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5
v_1	v_1	v_1	0	0	0	0	0	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5
0	0	0	v_2	v_2	v_2	v_2	v_2	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5
0	0	0	v_2	v_2	v_2	v_2	v_2	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5
0	0	0	v_2	v_2	v_2	v_2	v_2	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5
0	0	0	v_2	v_2	v_2	v_2	v_2	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5
0	0	0	v_2	v_2	v_2	v_2	v_2	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5
v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_3	v_3	v_3	v_3	0	0	0	0	0
v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_3	v_3	v_3	v_3	0	0	0	0	0
v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_3	v_3	v_3	v_3	0	0	0	0	0
v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_3	v_3	v_3	v_3	0	0	0	0	0
v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	0	0	0	0	v_4	v_4	v_4	v_4	v_4
v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	0	0	0	0	v_4	v_4	v_4	v_4	v_4
v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	0	0	0	0	v_4	v_4	v_4	v_4	v_4
v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	0	0	0	0	v_4	v_4	v_4	v_4	v_4
v_5	v_5	v_5	v_5	v_5	v_5	v_5	v_5	0	0	0	0	v_4	v_4	v_4	v_4	v_4

for a certain class. In this eigenvector, the values corresponding to instances of the same subclass of that class share the same value, while the values are different between subclasses and the values at the positions corresponding to other classes are 0.

Moreover, we can observe that eigenvectors showing the subclass structure of classes with a smaller number of instances correspond to bigger eigenvalues, and in the cases where multiple classes have an equal number of instances, eigenvectors that correspond to these classes merge. This means that in such eigenvectors structure of both classes can be observed, and the values corresponding to the other classes are zero. Such eigenvectors are repeated the number of times equal to the number of classes that have the same amount of elements.

In addition, we can observe that $\mathbf{L}_b$ is a constant sum block matrix [21], meaning that each of its rows and columns sums up to the same constant value. Being a constant sum block matrix, $\mathbf{L}_b$ is guaranteed to have a vector of ones as its eigenvector corresponding to eigenvalue 0 [21]. In addition, we can observe that for the data with a subclass structure, the eigenvectors maximizing the criterion (2.46) are those with the block structure as described.

$$\begin{pmatrix} r_1 & r_3 & 0 \\ r_1 & r_3 & 0 \\ r_1 & r_3 & 0 \\ r_1 & r_4 & 0 \\ r_1 & r_4 & 0 \\ r_1 & r_4 & 0 \\ r_1 & r_4 & 0 \\ r_1 & r_4 & 0 \\ r_2 & 0 & r_5 \\ r_2 & 0 & r_5 \\ r_2 & 0 & r_5 \\ r_2 & 0 & r_5 \\ r_2 & 0 & r_6 \\ r_2 & 0 & r_6 \\ r_2 & 0 & r_6 \\ r_2 & 0 & r_6 \\ r_2 & 0 & r_6 \end{pmatrix} \begin{matrix} c=1, z=1 \\ c=1, z=1 \\ c=1, z=1 \\ c=1, z=2 \\ c=1, z=2 \\ c=1, z=2 \\ c=1, z=2 \\ c=1, z=2 \\ c=2, z=1 \\ c=2, z=1 \\ c=2, z=1 \\ c=2, z=1 \\ c=2, z=2 \\ c=2, z=2 \\ c=2, z=2 \\ c=2, z=2 \\ c=2, z=2 \end{matrix} . \tag{3.2}$$

Due to the described properties of $\mathbf{L}_b$, its eigendecomposition can be avoided and substituted with a much faster process: first, we select a vector of ones as its first eigenvector and then create $Z * C - 1$ eigenvectors of random values following the structure as described earlier, and orthogonalize them following the Gram-Shmidt process [22]. In the end, we remove the vector of ones since it is useless as it maps data instances to the same points. The process is described in detail in Algorithm 1.

Algorithm 1: Target vectors calculation, single-view case

Function `getSingleviewTargets(`$class\_labels$`,`$cluster\_labels$`,`C`,`Z`,`N`):`

Input: $class\_labels : N \times 1$ vector with class labels;

$cluster\_labels : N \times 1$ vector with the cluster labels;

Z : number of clusters in each class;

C : number of classes;

N : number of elements;

`%class-level vectors;`

$T \leftarrow N \times (C-1)$ matrix with random values at positions of different classes, such that values are repeated within the class in one column, but distinct between classes and columns;

$L \leftarrow$ unique numbers of elements in each class sorted in ascending order;

`%cluster level vectors;`

for $l \leftarrow$ iterate through *L* **do**

  $k \leftarrow$ list of classes with l elements;

  $m \leftarrow$ length(k);

  $Tclust \leftarrow N \times m * (Z-1)$ matrix with random values at positions of all subclasses of classes in k, such that the values are shared within the subclass in one column, but distinct between subclasses and columns. Values at positions of other classes are 0s;

  $T \leftarrow$ append $Tclust$ as columns on the right;

end

$T \leftarrow$ append $N \times 1$ vector of ones as a column on the left;

Orthogonalize T;

remove the first column of T;

return T

3.3 Multi-view Subclass Discriminant Analysis

As mentioned in the previous chapter, machine learning methods can take advantage of projecting multi-view data to some latent space and performing further analysis there. However, data within each view can be multi-modal as well, while most of the proposed methods rely on the assumption of its unimodality. To address this issue, a multi-view extension to Subclass Discriminant Analysis, the Multiview Subclass Discriminant Analysis (MvSDA), is proposed.

We seek to find a projection space, projection onto which would result in different classes being far from each other while keeping all data samples close to each other. To achieve this, we maximize the distance between the means of subclasses of different classes, while minimizing the total scatter of the mean-centered data. The total scatter is therefore defined as

$$\mathbf{S}_t = \sum_{i=1}^{V}\sum_{k=1}^{N} \mathbf{y}_k^i \mathbf{y}_k^{i\,T} = \mathbf{Y}\mathbf{Y}^T = \mathbf{W}^T\mathbf{X}\mathbf{X}^T\mathbf{W}, \tag{3.3}$$

where $\mathbf{y}_k^i$ is the $k'th$ sample of view i in the latent space. The between-class scatter matrix is defined as

$$\begin{aligned}\mathbf{S}_b &= \sum_{i=1}^{V}\sum_{j=1}^{V}\sum_{p=1}^{C}\sum_{\substack{q=1\\q\neq p}}^{C}\sum_{l=1}^{d_p}\sum_{h=1}^{d_q} p_{pl}^i p_{qh}^j (\boldsymbol{\mu}_{pl}^i - \boldsymbol{\mu}_{qh}^j)(\boldsymbol{\mu}_{pl}^i - \boldsymbol{\mu}_{qh}^j)^T \\ &= \sum_{i=1}^{V}\sum_{j=1}^{V} \mathbf{W}_i^T \mathbf{X}_i \mathbf{L}_{bij}^{mv} \mathbf{X}_j^T \mathbf{W}_j = \mathbf{W}^T \mathbf{X}\mathbf{L}_b^{mv}\mathbf{X}^T\mathbf{W},\end{aligned} \tag{3.4}$$

$$\mathbf{X} = \begin{pmatrix} \mathbf{X}_1 & 0 & ... & 0 \\ 0 & \mathbf{X}_2 & ... & 0 \\ 0 & 0 & ... & \mathbf{X}_V \end{pmatrix}, \tag{3.5}$$

$$\mathbf{W} = \begin{pmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \\ ... \\ \mathbf{W}_V \end{pmatrix}, \tag{3.6}$$

$$\mathbf{L}_b^{mv} = \begin{pmatrix} \mathbf{L}_{b11}^{mv} & \mathbf{L}_{b21}^{mv} & ... & \mathbf{L}_{bV1}^{mv} \\ \mathbf{L}_{b12}^{mv} & \mathbf{L}_{b22}^{mv} & ... & \mathbf{L}_{bV2}^{mv} \\ ... & ... & ... & ... \\ \mathbf{L}_{b1V}^{mv} & \mathbf{L}_{b2V}^{mv} & ... & \mathbf{L}_{bVV}^{mv} \end{pmatrix}, \tag{3.7}$$

$$\mathbf{L}_{bij}^{mv} = \begin{cases} 2\sum_{p=1}^{C}\sum_{\substack{q=1 \\ q\neq p}}^{C}\sum_{l=1}^{d_p}\sum_{h=1}^{d_q}\frac{VN_{qh}^j}{N_{pl}^i N^2}\mathbf{e}_{pl}^i{\mathbf{e}_{pl}^i}^T - \frac{1}{N^2}\mathbf{e}_{pl}^i{\mathbf{e}_{qh}^j}^T, & \text{if } i=j \\ -2\sum_{p=1}^{C}\sum_{\substack{q=1 \\ q\neq p}}^{C}\sum_{l=1}^{d_p}\sum_{h=1}^{d_q}\frac{1}{N^2}\mathbf{e}_{pl}^i{\mathbf{e}_{qh}^j}^T, & \text{otherwise} \end{cases}, \tag{3.8}$$

where i and j are view labels, p and q are class labels, l and h are subclass labels, $p_{pl}^i = \frac{N_{pl}^i}{N}$ is the prior of the subclass l of class p in the view i, $\boldsymbol{\mu}_{pl}^i$ is the mean of the subclass l of class p in view i, $\mathbf{e}_{pl}^i$ is the vector of length N with ones at positions corresponding to subclass l of class p in view i and zeros elsewhere.

The optimal projection space is then obtained by optimizing the Fisher-Rao's criterion:

$$\mathcal{J}(\mathbf{W}) = \underset{\mathbf{W}_i^T\mathbf{W}_i=\mathbf{I}, i=1,...,V}{argmax} \frac{Tr(\mathbf{W}^T\mathbf{X}\mathbf{L}_b\mathbf{X}^T\mathbf{W})}{Tr(\mathbf{W}^T\mathbf{X}\mathbf{X}^T\mathbf{W})}, \tag{3.9}$$

where $\mathbf{X}$ and $\mathbf{W}$ are defined as in (3.5) and (3.6), respectively, and $\mathbf{K}$ is centered. Equivalently, solution to the kernel version of the method is obtained by optimizing

$$\mathcal{J}(\mathbf{A}) = \underset{\mathbf{A}^T\mathbf{K}\mathbf{A}=\mathbf{I}}{argmax} \frac{Tr(\mathbf{A}^T\mathbf{K}\mathbf{L}_b\mathbf{K}^T\mathbf{A})}{Tr(\mathbf{A}^T\mathbf{K}\mathbf{K}^T\mathbf{A})}, \tag{3.10}$$

where $\mathbf{K}$ is a block-diagonal matrix having $\mathbf{K}_v$ as its $v'th$ block:

$$K = \begin{pmatrix} \mathbf{K}_1 & 0 & ... & 0 \\ 0 & \mathbf{K}_2 & ... & 0 \\ 0 & 0 & ... & \mathbf{K}_V \end{pmatrix} \tag{3.11}$$

The solution to (3.9) is obtained by solving the eigendecomposition problem $\mathbf{L}_b\mathbf{X}^T\mathbf{v} = \lambda\mathbf{X}^T\mathbf{v}$. Similarly, the solution to (3.10) is given by $\mathbf{L}_b\mathbf{K}\mathbf{a} = \lambda\mathbf{K}\mathbf{a}$. The solution can be obtained following Spectral Regression similarly to the process described in section 3.1.

3.4 Speeding Up the Eigendecomposition Step: Multi-view Case

The proposed criterion for MvSDA can be solved by following a fast process similar to the one defined for single-view case, based on the structure of the between-class Laplacian matrix $\mathbf{L}_b^{mv}$. Let us consider an example of multi-view data of two views and two classes, where each class is represented by two subclasses. Let the first subclass of the first class in the first view contain 2 samples and the second subclass of the first class 2 samples as well. Let the first subclass of the second class contain 3 samples and second subclass 4 samples. In the second view, let the first subclass of the first class contain 2 samples and the second subclass 2 samples. The second class contains 4 samples in the first subclass and 3 samples in the second subclass.

In this case, the between-class Laplacian matrix has the structure outlined in Tab. 3.2, where v_1, v_2, v_3, v_4, v_5, v_6, v_7, and v_8 are the positive constant values, and v_9 is the constant negative value corresponding to positions that correspond to instances of different classes. The structure of the eigenvectors corresponding to this Laplacian matrix is outlined in (3.12), where c, z, and v are the class, subclass, and view labels, respectively, and r_i is the $i'th$ random value.

Similarly to the single-view case, we can notice that $\mathbf{L}_b^{mv}$ is a constant sum block matrix, having a vector of ones as its eigenvector, corresponding to the eigenvalue zero. From (3.2), we can observe the block structure of the matrix, and notice that the bigger blocks correspond to different views, and the structure within each diagonal view block is similar to that of the single-view case. In the inter-view blocks we can see the blocks corresponding to different classes. Due to this block structure, we can observe the similar structure in the eigenvectors of $\mathbf{L}_b^{mv}$.

Here and further we assume that all classes have the same number of subclasses equal to Z. Then, for the problem of C classes and V views, the rank of $\mathbf{L}_b^{mv}$ is equal to $C * Z * V - 1$, resulting in the same amount of nonzero eigenvalues.

The first $C-1$ eigenvectors have class block structure similar to the one in the single-view case, and the block structure is repeated across the positions corresponding to different views. The rest of the eigenvectors show the subclass structure of different classes across all the views, similarly to the single-view case. For a certain class, eigenvector showing its subclass structure, has the same constant value at positions corresponding to instances of the same subclass in one view and the values are different between subclasses. The values corresponding to the rest of the classes are 0s.

Similarly to the single-view case, we observe that the eigenvectors corresponding to classes having an equal number of elements merge together, and such eigenvectors are repeated as many times as there are merged classes. In addition to that, the classes with a smaller number of elements correspond to bigger eigenvalues.

Table 3.2. *Structure of the between-class Laplacian matrix in multi-view SDA.*

v_1	v_1	0	0	v_9	v_9	v_9	v_9	v_9	v_9	v_9	0	0	0	0	v_9	v_9	v_9	v_9	v_9	v_9	v_9
v_1	v_1	0	0	v_9	v_9	v_9	v_9	v_9	v_9	v_9	0	0	0	0	v_9	v_9	v_9	v_9	v_9	v_9	v_9
0	0	v_2	v_2	v_9	v_9	v_9	v_9	v_9	v_9	v_9	0	0	0	0	v_9	v_9	v_9	v_9	v_9	v_9	v_9
0	0	v_2	v_2	v_9	v_9	v_9	v_2	v_9	v_9	v_9	0	0	0	0	v_9	v_9	v_9	v_9	v_9	v_9	v_9
v_9	v_9	v_9	v_9	v_3	v_3	v_3	0	0	0	0	v_9	v_9	v_9	v_9	0	0	0	0	0	0	0
v_9	v_9	v_9	v_9	v_3	v_3	v_3	0	0	0	0	v_9	v_9	v_9	v_9	0	0	0	0	0	0	0
v_9	v_9	v_9	v_9	v_3	v_3	v_3	0	0	0	0	v_9	v_9	v_9	v_9	0	0	0	0	0	0	0
v_9	v_9	v_9	v_9	0	0	0	v_4	v_4	v_4	v_4	v_9	v_9	v_9	v_9	0	0	0	0	0	0	0
v_9	v_9	v_9	v_9	0	0	0	v_4	v_4	v_4	v_4	v_9	v_9	v_9	v_9	0	0	0	0	0	0	0
v_9	v_9	v_9	v_9	0	0	0	v_4	v_4	v_4	v_4	v_9	v_9	v_9	v_9	0	0	0	0	0	0	0
v_9	v_9	v_9	v_9	0	0	0	v_4	v_4	v_4	v_4	v_9	v_9	v_9	v_9	0	0	0	0	0	0	0
0	0	0	0	v_9	v_9	v_9	v_9	v_9	v_9	v_9	v_5	v_5	0	0	v_9	v_9	v_9	v_9	v_9	v_9	v_9
0	0	0	0	v_9	v_9	v_9	v_9	v_9	v_9	v_9	v_5	v_5	0	0	v_9	v_9	v_9	v_9	v_9	v_9	v_9
0	0	0	0	v_9	v_9	v_9	v_9	v_9	v_9	v_9	0	0	v_6	v_6	v_9	v_9	v_9	v_9	v_9	v_9	v_9
0	0	0	0	v_9	v_9	v_9	v_9	v_9	v_9	v_9	0	0	v_6	v_6	v_9	v_9	v_9	v_9	v_9	v_9	v_9
v_9	v_9	v_9	v_9	0	0	0	0	0	0	0	v_9	v_9	v_9	v_9	v_7	v_7	v_7	v_7	0	0	0
v_9	v_9	v_9	v_9	0	0	0	0	0	0	0	v_9	v_9	v_9	v_9	v_7	v_7	v_7	v_7	0	0	0
v_9	v_9	v_9	v_9	0	0	0	0	0	0	0	v_9	v_9	v_9	v_9	v_7	v_7	v_7	v_7	0	0	0
v_9	v_9	v_9	v_9	0	0	0	0	0	0	0	v_9	v_9	v_9	v_9	v_7	v_7	v_7	v_7	0	0	0
v_9	v_9	v_9	v_9	0	0	0	0	0	0	0	v_9	v_9	v_9	v_9	0	0	0	0	v_8	v_8	v_8
v_9	v_9	v_9	v_9	0	0	0	0	0	0	0	v_9	v_9	v_9	v_9	0	0	0	0	v_8	v_8	v_8
v_9	v_9	v_9	v_9	0	0	0	0	0	0	0	v_9	v_9	v_9	v_9	0	0	0	0	v_8	v_8	v_8

Hence, we can follow a similar procedure of creation of eigenvectors: we select the vector of ones as our first eigenvector, and obtain the rest by creating vectors of random values following the described structure and orthogonalizing them following the Gram-Schmidt process. The vector of ones is then removed as being useless. The process is described in more details in Algorithm 2.

$$
\begin{array}{c}
\\
\begin{pmatrix}
r_1 & r_5 & r_9 & r_{13} & 0 & 0 & 0 \\
r_1 & r_5 & r_9 & r_{13} & 0 & 0 & 0 \\
r_1 & r_6 & r_{10} & r_{14} & 0 & 0 & 0 \\
r_1 & r_6 & r_{10} & r_{14} & 0 & 0 & 0 \\
r_2 & 0 & 0 & 0 & r_{17} & r_{21} & r_{25} \\
r_2 & 0 & 0 & 0 & r_{17} & r_{21} & r_{25} \\
r_2 & 0 & 0 & 0 & r_{17} & r_{21} & r_{25} \\
r_2 & 0 & 0 & 0 & r_{18} & r_{22} & r_{26} \\
r_2 & 0 & 0 & 0 & r_{18} & r_{22} & r_{26} \\
r_2 & 0 & 0 & 0 & r_{18} & r_{22} & r_{26} \\
r_2 & 0 & 0 & 0 & r_{18} & r_{22} & r_{26} \\
r_3 & r_7 & r_{11} & r_{15} & 0 & 0 & 0 \\
r_3 & r_7 & r_{11} & r_{15} & 0 & 0 & 0 \\
r_3 & r_8 & r_{12} & r_{16} & 0 & 0 & 0 \\
r_3 & r_8 & r_{12} & r_{16} & 0 & 0 & 0 \\
r_4 & 0 & 0 & 0 & r_{19} & r_{23} & r_{27} \\
r_4 & 0 & 0 & 0 & r_{19} & r_{23} & r_{27} \\
r_4 & 0 & 0 & 0 & r_{19} & r_{23} & r_{27} \\
r_4 & 0 & 0 & 0 & r_{19} & r_{23} & r_{27} \\
r_4 & 0 & 0 & 0 & r_{20} & r_{24} & r_{28} \\
r_4 & 0 & 0 & 0 & r_{20} & r_{24} & r_{28} \\
r_4 & 0 & 0 & 0 & r_{20} & r_{24} & r_{28}
\end{pmatrix}
\end{array}
\begin{array}{ccc}
c & z & v \\
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 2 & 1 \\
1 & 2 & 1 \\
2 & 1 & 1 \\
2 & 1 & 1 \\
2 & 1 & 1 \\
2 & 2 & 1 \\
2 & 2 & 1 \\
2 & 2 & 1 \\
2 & 2 & 1 \\
1 & 1 & 2 \\
1 & 1 & 2 \\
1 & 2 & 2 \\
1 & 2 & 2 \\
2 & 1 & 2 \\
2 & 1 & 2 \\
2 & 1 & 2 \\
2 & 1 & 2 \\
2 & 2 & 2 \\
2 & 2 & 2 \\
2 & 2 & 2
\end{array}
. \tag{3.12}
$$

Algorithm 2: Target vectors calculation, multi-view case

Function `getMultiviewTargets(`$class\_labels$,$cluster\_labels$,V,C,Z,N`)`**:**

Input: $class\_labels : V * N \times 1$ vector with class labels;

$cluster\_labels : V * N \times 1$ vector with the cluster labels;

V : number of views;

Z : number of clusters in each class;

C : number of classes; N : number of elements;

`%class-level vectors;`

$T \leftarrow V * N \times (C - 1)$ matrix with random values at positions of different classes, such that values are repeated within the class in one column, but distinct between views, classes, and columns;

$L \leftarrow$ unique numbers of elements in each class sorted in ascending order;

`%cluster level vectors;`

for $l \leftarrow$ iterate through *L* **do**

  $k \leftarrow$ list of classes with l elements;

  $m \leftarrow$ length(k);

  $Tclust \leftarrow V * N \times m * (V * Z - 1)$ matrix with random values at positions of all subclasses of classes in k, such that the values are shared within the subclass in one column, but distinct between subclasses, views, and columns. Values at positions of other classes are 0s;

  $T \leftarrow$ append $Tclust$ as columns on the right;

end

$T \leftarrow$ append $N \times 1$ vector of ones as a column on the left ;

Orthogonalize T;

remove the first column of T;

return T

4 EXPERIMENTAL EVALUATION

In this chapter, the experiments that were performed in order to evaluate the proposed extensions are presented. The approaches are evaluated on nine single-view datasets and seven multi-view datasets, but due to resampling of some of the multi-view datasets, we obtain nine subsets for the multi-view case as well. In single-view methods, the proposed speed-up approach is compared with several clustering-based approaches that rely on eigendecomposition, namely, SDA, CDA, and SMFA. Comparison is done also with Spectral Regression Discriminant Analysis (SRDA). In addition, the results are compared with Spectral Regression Subclass Discriminant Analysis in order to evaluate the performance obtained by creating target vectors using the proposed approach based on eigendecomposition. In the multi-view case, the proposed approach is compared with other multi-view methods, namely SMvDA and MvMDA, and the proposed single-view approach, where the features from different views are concatenated.

In the experiments, we assume that the subclass labels of all instances are known, and we obtain them with k-means clustering in $\mathbb{R}^D$. For the multi-view datasets, clustering is performed in each view separately. Same subclass labels are used for all the methods, and for the kernel formulations of the algorithms. For the kernel formulations, we exploit the Gaussian RBF kernel function:

$$\mathbf{K}(x_i, x_j) = exp(-\frac{||\mathbf{x}_i - \mathbf{x}_j||_2^2}{2\sigma^2}), \tag{4.1}$$

where we set the Gaussian scale σ to the mean Euclidean distance between the vectors.

The dimensionality of the projection space is determined by the rank of the between-class Laplacian matrix $\mathbf{L}_b$ or $\mathbf{L}_b^{mv}$, equal to $C * Z - 1$ and $V * C * Z - 1$, respectively, where V is the number of views, C is the number of classes, and Z is the number of subclasses in each class. After projection, classification is done with k-Nearest Neighbors algorithm with k=5.

All the hyperparameters are selected with grid search. These include the k_{Int} and k_{Pen} parameters in SMFA and KSMFA, that were selected from the range of [2..14] with step 3 for k_{Int} and [20..100] with step 20 for k_{Pen}. The regularization parameter in SRDA was selected from the set of $\{10^{-4}, 10^{-3}, 10^{-2}, 0, 1, 10^1, 10^2\}$. Regularization was applied to all kernel single-view methods and all multi-view methods with regularization parameter selected from the same set. For efficient matrix inversion, Cholesky decomposition was used for efficient matrix inversion.

Approximate kernel regression was used for large datasets of over 2500 samples, and prototypes were selected by selecting 1500 random vectors from the training data. For kernel SDA, SMFA, and CDA, Nyström-based approximate kernel regression was used with cardinality of 1000, due to infeasible computational requirements of eigendecomposition of the full matrix.

5-fold stratified cross-validation was applied to obtain the training, testing, and validation splits. 60% of the data was selected for training, 20% - for validation, i.e., hyperparameter tuning, and 20% - for testing. The results are reported for the models trained on the training set and tested on the testing set. All experiments were performed on a computer with 4-core Intel i7-4800Q CPU and 32 GB of RAM.

4.1 Single-view Datasets

In order to validate the proposed approach, we perform experiments on nine single-view datasets. One of the datasets is a large-scale facial recognition dataset, four are facial image datasets and the rest are datasets of different domains, used to prove the applicability of the proposed approach on various data types.

The Jaffe dataset [39] consists of images of ten Japanese female models with seven different facial expressions: anger, happiness, fear, disgust, sadness, surprise and neutral. The dataset includes 213 images. The example image can be seen in Fig. 4.1.

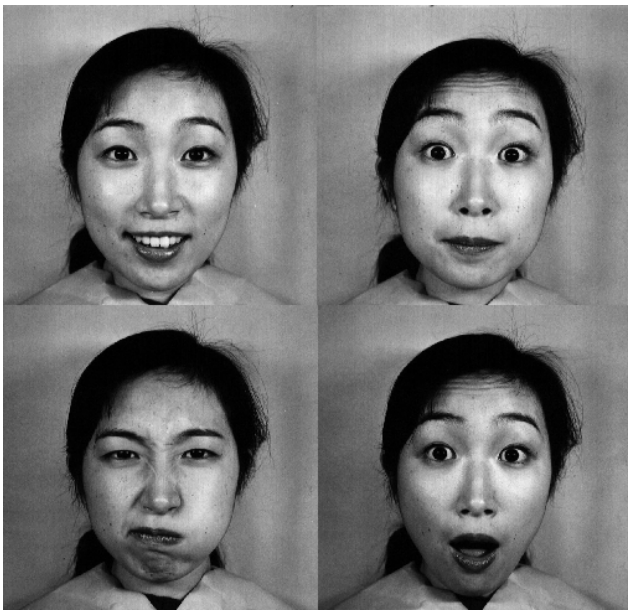

Figure 4.1. *Example of images from Jaffe dataset.*

The second facial image dataset is the BU dataset [62], containing 700 samples and 7 facial expression classes. Another facial image dataset is the extended Cohn-Kanade dataset [33], containing the same 7 facial expressions and 245 images, example of which can be seen in Fig. 4.2.

Figure 4.2. *Example of images from Cohn-Kanade dataset.*

The Extended Yale-B [36] dataset poses the problem of face recognition of 38 individuals, where each individual has 64 images taken under different positions, view angles, illumination conditions, etc. The dataset contains 2432 grayscale images. An example can be seen in Fig. 4.3.

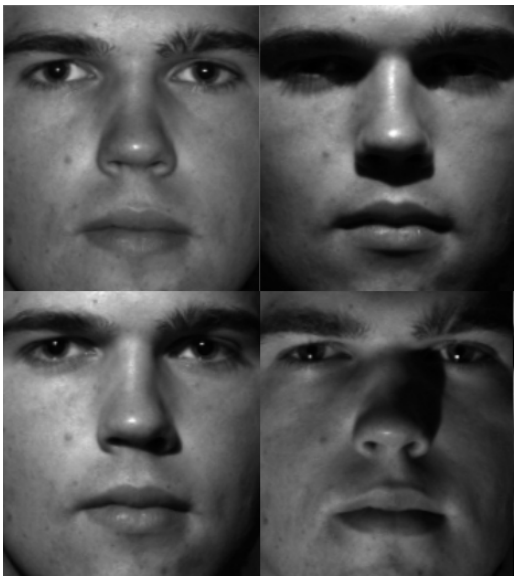

Figure 4.3. *Example of images from Extended Yale-B dataset.*

In order to evaluate the proposed approach on the large-scale problems, we utilize one large-scale facial image dataset. The SoF dataset [2] consists of images of 66 male and 4 female people, resulting in a total of 42,592 images. The images were taken under various illumination conditions and various occlusions, for example, glasses. Example images from the SoF dataset can be seen in Fig. 4.4. In all the facial image datasets mentioned above, the features were obtained by rescaling the image to 30×40 pixel size and flattening to obtain a 1×1200 vector.

Figure 4.4. *Example of images from SoF dataset.*

Other datasets used in this work are meant to show the applicability of the methods on various types of data. The Ionosphere dataset [53] poses the problem of classification of radar data sequences to the ones containing certain evidence of a certain type of structure in the ionosphere and the ones containing no evidence, hence defining a binary classification problem. Data is represented by radar measurements, where each sample is a 34-dimensional vector and there are 351 samples in the dataset.

Another application area evaluated in this work is the recognition of handwritten digits. For this purpose, the Semeion dataset [6] was used. The dataset contains 1593 images of handwritten digits produced by 80 people. Each person has written each digit twice: with a normal handwriting style and with fast handwriting. The images are binarized and rescaled to 16x16 pixels, and further flattened to obtain 1x256 vectors.

The MONKS-2 dataset [58] is a classical dataset that was initially proposed as a benchmark for evaluation of different learning algorithms, the subject area of which is an artificially constructed area of robot recognition based on their features. Each sample of the data is represented by 6 discrete features and there are 2 classes in the dataset.

The Pima Indians Diabetes dataset [54] poses the problem of diabetes diagnosis based on other health attributes of the patient, therefore defining a binary classification problem. The dataset has information on 768 patients, along with the information on the number of pregnancies the patient had, BMI, insulin level, age. Table 4.1 outlines the summary of all single-view datasets used in this work.

4.2 Multi-view Datasets

In order to evaluate the proposed Multi-view Subclass Discriminant Analysis approach, experiments on nine multi-view datasets were conducted. In order to show the applicability of the proposed approach on various data types, the selected datasets include several human action recognition datasets using various sensors, visual object classification datasets, handwritten digit recognition dataset, and one audio classification dataset [37].

Table 4.1. *Summary of single-view datasets.*

Dataset	Dimensionality	Number of classes	Number of samples	Subject area
Jaffe	1200	7	213	facial expression recognition
Cohn-Kanade	1200	7	245	facial expression recognition
BU	1200	7	700	facial expression recognition
Yale-B	1200	20	2432	object recognition
Ionosphere	34	2	351	radar data analysis
Semeion	256	10	1593	handwritten digits recognition
MONKS-2	6	2	169	robot recognition (artificially created)
Pima	8	2	768	action recognition
SoF	1200	112	42592	face recognition

The first dataset used is the Handwritten digits dataset [5]. The dataset poses the problem of image-based handwritten digit recognition from multiple views, therefore defining a multiclass classification problem with 10 classes. Each sample is described from 6 different views: Fourier coefficients of length 128, Karhunen-Love coefficients of length 64, Zernike moments of length 47, morphological features of length 6 and profile correlations of length 76. The dataset consists of 2000 samples extracted from a collection of Dutch utility maps. Prior to extraction of the features, the images were binarized.

The Caltech-101 dataset [18] poses a problem of image-based object classification based on the features extracted from six views: Histogram of Oriented Gradients features of length 1984, GIST features of length 512, Gabor features of length 48, wavelet moments of length 40, CENTRIST features of length 254 and local binary patterns features of length 1928. The classes of the initial dataset are largely imbalanced, causing the creation of two distinct datasets with 7 and 20 classes each, further referred to as Caltech-101-7 and Caltech-101-20. Caltech-101-7 contains 1474 images of 7 classes: face, motorbikes, dollar bill, garfield, snoopy, stop sign and windsor chair. Caltech-101-20 contains 2386 images of 20 classes: face, leopards, motorbikes, binocular, brain, camera, car side, dollar bill, ferry, garfield, hedgehog, pagoda, rhino, snoopy, stapler, stop sign, water lilly, windsor chair, wrench, and yin-yang. Several example images can be seen in FIg. 4.5.

Another object recognition dataset is the NUS-WIDE dataset [14] containing images of 31 classes: bear, bird, boat, book, car, cat, computer, coral, cow, dog, elk, fish, flags, flower,

Figure 4.5. *Example of images from Caltech-101 dataset.*

fox, horse, leaf, plane, rocks, sand, sign, statue, sun, tiger, tower, toy, train, tree, vehicle, whales, and zebra. The subset of 11288 samples was selected for the experiments, preserving the class balance. Each sample is described from 5 different views: wavelet texture of length 129, edge distribution of length 74, color correlation of length 145, color moments of length 226 and color histogram of length 65. Several example images from the dataset can be seen in Fig. 4.6.

Figure 4.6. *Example of images from NUS-WIDE dataset.*

In order to evaluate the proposed method on action recognition problems, one of the selected datasets is the Human Action Recognition Using Smartphones (HARS) dataset

(HARS) [4]. This dataset contains data collected from the accelerometer and gyroscope of the smartphone attached to the waist of a person. The person is performing one of the activities: walking, walking upstairs, walking downstairs, sitting, standing, lying. The data was collected from 20 people of various age groups in the range of 19-48 years. The 3-axial angular velocity and acceleration are captured at the rate of 50 Hz and further processed by applying denoising filters and sampling with a sliding window of 2.56 seconds and 50% overlap. Using Butterworth filter with 0.3Hz cutoff, the acceleration signal was separated into gravitational and body motion components, under the assumption that the gravitational component contains only low-frequency components. Therefore, the obtained dataset contains data from 9 views: angular velocity of each of 3 axes, total acceleration of each of 3 axes and body acceleration of each axis. The dataset contains 7352 samples and the cross-validation sets were selected in a way that the people performing the actions are not repeated between the train and test splits.

Another action recognition dataset targets the problem of action recognition of older people. The Healthy Old People Action Recognition dataset (HOPAR) [56] contains two independent subsets. The data was collected from a wireless sensor worn by a person and 4 activities were observed: sitting on a chair, sitting on a bed, ambulating and lying. Each sample is described from 4 views: acceleration of each of the 3 axes and the signal attributes, containing RSSI, frequency, and phase. In the first subset, data was collected from 60 people and 10495 samples were selected randomly, preserving the balance between the classes and the subjects. In the second subset, 27 people participated in the experiments resulting in 9057 samples in the dataset. Similarly to the HARS dataset, cross-validation splits were selected in such a way that the subjects are not repeated between the train and test sets.

The Robot Execution Failures dataset [1] is often used for evaluation of the methods applied to robotics problems. Here, the data consists of 5 subsets, each describing a different problem. For our experiments, datasets 1 and 4 were combined, resulting in a dataset of failures in approach to grasp or ungrasp position by a robot, therefore defining a multiclass classification problem with 4 classes: normal, collision, frontal collision, and obstruction. Each sample in the dataset is described from 6 views: force on each of the 3 axes and torque on each of the 3 axes. There are a total of 205 samples in the dataset.

Another application area that is evaluated in our experiments is audio processing. Here, we tackle a problem of genre classification utilizing the Million Song Dataset with Images (MSDI) [43]. Here, the task is to classify audio samples into 15 different genres: blues, country, electronic, folk, jazz, latin, metal, newAge, pop, punk, rap, reggae, RnB, rock. Each audio sample is accompanied by an image of the corresponding album cover, described with the features extracted from CNN. Due to the large volume of the dataset, the subset of 7468 instances is selected for our experiments, preserving the balance between classes.

The summary of the multi-view datasets is outlined in Table 4.2.

Table 4.2. *Summary of multi-view datasets.*

Dataset	Number of views	Number of classes	Number of samples	Subject area
Handwritten digits	6	10	2000	handwritten digits recognition
Caltech-101-7	6	7	1474	object recognition
Caltech-101-20	6	20	2386	object recognition
NUS-WIDE	5	31	11288	object recognition
Robots Failures	6	4	205	robotics
HARS	9	6	7352	action recognition
HOPAR1	4	4	10495	action recognition
HOPAR2	4	4	9057	action recognition
MSDI	2	15	7468	genre classification

4.3 Results

The classification results for the single-view datasets can be seen in Table 4.3 and Table 4.4 for the linear formulations of the algorithms; and Table 4.5 and Table 4.6 for the kernelized formulations of the algorithms. As mentioned previously, the proposed approach is compared with SDA, CDA, SMFA, and SRDA. In addition, by performing eigendecomposition of L_b, regressing the obtained eigenvectors, projecting the data to the obtained vector and sorting the vectors according to calculated criterion value, we verify that for the data with subclass structure the eigenvectors corresponding to larger criterion values are those following the described structure.

For all the experiments, we report the accuracy, training time, and the number of subclasses that resulted in the best accuracy after testing on the test set with 1-6 subclasses. In the single-view dataset, the training time excludes the time taken for clustering, as a comparison is done with other clustering-based algorithms, hence, this time is similar for all the methods. In the multi-view case, the clustering time is included in the total time reported for the proposed method, as the comparison is done with the methods, that do not require clustering.

As can be seen, the proposed speed-up approach results in higher speed and competitive accuracy. We can see that in the linear case, the proposed method outperformed other methods on 6 out of 9 datasets while being close to the best accuracy on the rest 3 datasets. In terms of speed, the method is comparable with SRDA, that follows a similar speed-up approach, but results in better accuracy even for the single subclass case.

In the multi-view datasets, the comparison is done with multi-view extensions of LDA,

Table 4.3. *Classification results of linear methods in single-view datasets: accuracy/number of clusters per class.*

Dataset	SDA	CDA	SMFA	SRDA	SDA, sorted vectors	fastSDA (our)
BU	62.8/1	60.1/1	59.9/1	62.6	**63.3/1**	**63.3/1**
Jaffe	65.2/1	58.1/1	63.8/1	65.7	65.2/1	**66.2/1**
Ionos.	**89.7/3**	89.4/5	89.4/4	83.1	87.8/6	88.3/2
Kanade	63.3/1	61.6/1	55.1/1	65.3	64.0/1	**65.7/1**
Semeion	87.8/1	83.2/1	86.7/1	88.9	89.0/1	**89.4/1**
Yale	86.8/2	86.6/2	87.6/2	88.6	88.7/1	**89.4/1**
PIMA	71.2/5	72.0/5	**72.8/2**	71.2	71.2/1	71.6/4
Monks2	55.8/2	53.9/1	**61.2/1**	50.9	58.8/6	52.7/3
SoF	98.6/1	98.9/1	98.5/1	**99.0**	98.0/1	**99.0/1**

Table 4.4. *Classification results of linear methods in single-view datasets: training time (in sec).*

Dataset	SDA	CDA	SMFA	SRDA	SDA, sorted vectors	fastSDA (our)
BU	0.019	0.017	0.030	0.013	0.09	**0.005**
Jaffe	0.013	0.004	0.005	0.005	0.013	**0.002**
Ionos.	0.008	**0.002**	0.005	0.005	0.017	**0.002**
Kanade	0.012	0.005	0.006	0.005	0.02	**0.002**
Semeion	0.245	0.041	0.147	0.015	1.148	**0.013**
Yale	0.063	0.056	0.216	0.010	4.1	**0.007**
PIMA	0.003	0.009	0.016	0.005	0.081	**0.001**
Monks2	0.004	0.002	0.002	0.005	0.005	**0.001**
SoF	9.52	18.3	86.0	0.831	0.801	**0.611**

namely SMvDA, MvMDA, and the proposed single-view approach, where the features of the different views are concatenated and the clustering is berformed on the concatenated features. The results for the multi-view datasets are outlined in Table 4.7 and Table 4.8 for the linear versions of the algorithms; and in Table 4.9 and Table 4.10 for the kernelized forms of the algorithms.

As can be seen, in the linear case, the proposed mvSDA outperforms the multi-view LDA

Table 4.5. *Classification results of kernel methods in single-view datasets: accuracy/number of clusters per class.*

Dataset	kernel SDA	kernel CDA	kernel SMFA	kernel fastSDA (our)
BU	63.7/1	**64.7/1**	62.4/1	64.2/1
Jaffe	**69.0/1**	**69.0/1**	63.8/1	68.5/1
Ionosphere	83.4/6	94.5/5	84.6/3	**94.9/6**
Kanade	59.6/2	60.4/1	57.9/1	**61.2/1**
Semeion	91.2/2	91.5/1	**91.6/1**	90.6/1
Yale	89.4/6	**91.4/1**	75.2/4	**91.4/1**
PIMA	63.1/6	66.9/6	64.8/5	**72.3/3**
Monks2	46.0/6	**56.4/5**	55.2/2	52.7/3
SoF	77.4/2	79.2/2	98.3/2	**98.4/2**

Table 4.6. *Classification results of kernel methods in single-view datasets: training time (in sec).*

Dataset	kernel SDA	kernel CDA	kernel SMFA	kernel fastSDA (our)
BU	0.036	0.068	0.432	**0.01**
Jaffe	0.004	0.010	0.015	**0.001**
Ionosphere	0.012	0.026	0.049	**0.002**
Kanade	0.005	0.007	0.020	**0.001**
Semeion	0.275	0.479	11.5	**0.043**
Yale	1.00	0.886	39.5	**0.103**
PIMA	0.040	0.392	0.368	**0.012**
Monks2	0.02	0.127	0.007	**0.001**
SoF	188.9	190.3	167.7	**1.55**

extensions on all but one dataset. At the same time, in the kernel formulation, the proposed approach outperforms other methods on 4 datasets. The kernel formulation of the proposed single-view fastSDA outperforms other methods in 4 datasets. The proposed approaches also outperform the others in terms of speed by a large margin.

Table 4.7. *Classification results of linear methods in multi-view datasets: accuracy/number of clusters per class.*

Dataset	SMvDA	MvMDA	MvSDA (our)	single-view fastSDA (our)
HWD	**98.9**	98.6	98.8/1	98.5/4
HARS	62.6	31.9	**67.3/1**	63.0/3
Robots	66.8	57.5	**74.6/5**	46.4/6
Caltech-7	**98.2**	**98.2**	**98.2/1**	97.0/1
Caltech-20	93.7	94.6	**95.0/1**	89.7/1
HOPAR 1	84.9	84.8	**85.4/1**	84.8/2
HOPAR 2	81.9	81.9	**82.3/2**	**82.3/6**
MSDI	57.6	57.0	**58.4/6**	58.3/2
NUS-WIDE	48.6	47.3	**56.0/3**	26.0/3

Table 4.8. *Classification results of linear methods in multi-view datasets: training time (in sec).*

Dataset	SMvDA	MvMDA	MvSDA (our)	single-view fastSDA (our)
HWD	3.3	2.3	0.10	**0.03**
HARS	27.4	22.3	1.34	**0.22**
Robots	0.029	0.028	0.01	**0.002**
Caltech-7	21.5	22.4	1.35	**0.65**
Caltech-20	23.4	20.2	2.0	**1.0**
HOPAR 1	7.14	6.2	0.04	**0.008**
HOPAR 2	5.75	4.41	0.06	**0.008**
MSDI	58.4	50.9	0.2	**0.024**
NUS-WIDE	133.2	130.2	0.54	**0.09**

Table 4.9. *Classification results of kernel methods in multi-view datasets: accuracy/number of clusters per class.*

Dataset	kernel SMvDA	kernel MvMDA	kernel MvSDA (our)	kernel single-view fastSDA (our)
HWD	99.0	98.5	**99.3/1**	99.0/3
HARS	79.4	86.5	**89.5/3**	**89.5/2**
Robots	68.3	75.2	**81.5/2**	77.6/4
Caltech-7	97.6	**97.9**	97.7/1	97.8/1
Caltech-20	87.2	93.6	93.9/1	**94.7/1**
HOPAR 1	85.4	**86.0**	**86.0/2**	85.8/2
HOPAR 2	**83.1**	79.0	80.2/4	82.7/3
MSDI	51.3	31.6	61.5/1	**63.9/1**
NUS-WIDE	32.9	42	61.3/1	**62.7/1**

Table 4.10. *Classification results of kernel methods in multi-view datasets: training time (in sec).*

Dataset	kernel SMvDA	kernel MvMDA	kernel MvSDA (our)	kernel single-view fastSDA (our)
HWD	72.4	70.5	5.7	**0.07**
HARS	561	554	97	**4.3**
Robots	0.14	0.14	0.03	**0.001**
Caltech-7	30.9	30	2.78	**0.03**
Caltech-20	244	236	9.57	**0.11**
HOPAR 1	76.7	74.4	10.9	**4.49**
HOPAR 2	65.9	74.1	8.89	**2.8**
MSDI	69.9	48.6	**1.16**	2.3
NUS-WIDE	259.5	235.3	24	**7.7**

5 CONCLUSIONS

In this thesis, the area of dimensionality reduction by means of single-view and multi-view subspace learning was studied. The main limitations of the commonly used methods, including LDA, lie in the assumptions of the unimodality of data, limitations in the dimensionality of the learned subspace, that is limited by the rank of the between-class scatter matrix, and low training speed on high-dimensional data in linear case and large datasets in kernel case.

Many approaches have been proposed to overcome the first two of the above-mentioned limitations, including Subclass Discriminant Analysis, Clustering Discriminant Analysis, and Subclass Marginal Fisher Analysis, but these methods still suffer from limitations related to training time. In this work, an approach to overcome these three limitations at the same time was proposed by introducing a speed-up approach for Subclass Discriminant Analysis and Kernel Subclass Discriminant Analysis that is based on Spectral Regression and exploitation of the specific structure of the between-class Laplacian matrix. Experimentally it was shown that the proposed approach allows gaining the training speed while resulting in competitive performance.

At the same time, in the multi-view scenario, not as many approaches to overcoming the limitations of LDA were proposed. Therefore, a novel multi-view extension to Subclass Discriminant Analysis was proposed for the problems where data is represented by multiple modalities, along with a speed-up solution to it, that is closely related to the speed-up solution proposed for the single-view case.

Revisiting the research problems defined in the beginning of the work, it can be seen that the identifined requirements are satisfied in the proposed methods:

- the assumption of unimodality of each class is relaxed by relying on the subclass representation
- computational efficiency is achieved by substitution of the eigendecomposition step with a much faster process
- the possible dimensionality of the projected data is increased to $C * Z - 1$ or $C * Z * V - 1$ in single-view and multi-view cases, respectively
- the extention to multi-view data is proposed along with a fast solution.

Experimentally it was shown that the proposed approach outperforms other methods that rely on the assumption on the unimodality of the data, while taking much less time to train. By utilizing the proposed approaches, dimensionality reduction can be performed on the large-scale high-dimensional datasets, where the application of other methods is not viable; and the analysis can be performed on both single-view and multi-view data.

In the future work, possible research directions include the extensions that would rely on clustering in the kernel space for the kernelized formulation of the methods. Besides, estimation of the suitable amount of subclasses as a part of the training process can be studied.

REFERENCES

[1] M. Afifi and A. Abdelhamed. Integration and learning in supervision of flexible assembly systems. *IEEE Transactions on Robotics and Automation* 12 (1996), 202–219.

[2] M. Afifi and A. Abdelhamed. AFIF4: Deep gender classification based on an AdaBoost-based fusion of isolated facial features and foggy faces. *arXiv preprint arXiv:1706.04277* (2017).

[3] D. Amodei et al. Deep speech 2: End-to-end speech recognition in english and mandarin. *International Conference on Machine Learning*. (2016), 173–182.

[4] D. Anguita et al. A public domain dataset for human activity recognition using smartphones. *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*. Bruges, Belgium, (2013).

[5] M. van Breukelen, R. Duian, D. Tax and J. den Hartog. Handwritten digit recognition by combined classifiers. *Kybernetika* 34 (1998), 381–386.

[6] M. Buscema. MetaNet: the theory of independent judges. *Substance Use Misuse* 33 (1998), 439–461.

[7] D. Cai, X. He and J. Han. Spectral regression for efficient regularized subspace learning. *IEEE International Conference on Computer Vision*. Rio de Janeiro, Brazil, (2007).

[8] D. Cai, X. He and J. Han. SRDA: an efficient algorithm for large scale discriminant analysis. *IEEE Transactions on Knowledge and Data Engineering* 20 (2007), 1–12.

[9] D. Cai, X. He and J. Han. Speed up kernel discriminant analysis. *The VLDB Journal* 20 (2011), 21–33.

[10] G. Cao, A. Iosifidis, K. Chen and M. Gabbouj. Generalized multi-view embedding for visual recognition and cross-modal retrieval. *IEEE Transactions on Cybernetics* 48 (2018), 2542–2555.

[11] G. Cao, A. Iosifidis and M. Gabbouj. Multi-view nonparametric discriminant analysis for image retrieval and recognition. *IEEE Signal Processing Letters* 24.10 (2017), 1537–1541.

[12] B. Chen, L. Yuan, H. Liu and Z. Bao. Kernel subclass discriminant analysis. *Neurocomputing* 71 (2007), 455–458.

[13] X. Chen and T. Huang. Facial expression recognition: a clustering-based approach. *Pattern Recognition Letters* 24 (2003), 1295–1302.

[14] T. Chua et al. NUS-WIDE: A real-world web image database from National University of Singapore. *ACM International Conference on Image and Video Retrieval*. Greece, (2009).

[15] K. Chumachenko, J. Raitoharju, A. Iosifidis and M. Gabbouj. Speed-up and Multi-view Extensions to Subclass Discriminant Analysis. *arXiv preprint arXiv:1905.00794* (2019).

[16] R. Duda, P. Hart and D. Stork. *Pattern Classification*. 2nd. New York, NY, USA: Wiley, 2000.

[17] C. Feichtenhofer, A. Pinz and A. Zisserman. Convolutional two-stream network fusion for video action recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. (2016), 1933–1941.

[18] L. Fei-Fei, R. Fergus and P. Perona. One-Shot learning of object categories. *IEEE Transactions on Pattern Recognition and Machine Intelligence* 8 (2006), 594–611.

[19] R. Fisher. The statistical utilization of multiple measurements. *Annals of Eugenics* 8 (1938), 376–386.

[20] S. Hershey et al. CNN architectures for large-scale audio classification. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE. (2017), 131–135.

[21] S. Hill, M. Lettington and K. Schmidt. Block representation and spectral properties of constant sum matrices. *Electronic Journal of Linear Algebra* 34 (2018), 170–190.

[22] R. Horn and C. Johnson. *Matrix Analysis*. 1st. Cambridge University Press, 1985.

[23] A. Iosifidis and M. Gabbouj. Nyström-based approximate kernel subspace learning. *Pattern Recognition* 57 (2016), 190–197.

[24] A. Iosifidis and M. Gabbouj. Scaling up class-specific kernel discriminant analysis for large-scale face verification. *IEEE Transactions on Information Forensics and Security* 11 (2016), 2453–2465.

[25] A. Iosifidis and M. Gabbouj. Class-specific kernel discriminant analysis revisited: further analysis and extensions. *IEEE Transactions on Cybernetics* 47 (2017), 4485–4496.

[26] A. Iosifidis, A. Tefas and I. Pitas. On the optimal class representation in linear discriminant analysis. *IEEE Transactions on Neural Networks and Learning Systems* 24.9 (2013), 1491–1497.

[27] A. Iosifidis, A. Tefas and I. Pitas. Kernel reference discriminant analysis. *Pattern Recognition Letters* 49 (2014), 85–91.

[28] A. Iosifidis, A. Tefas and I. Pitas. Class-specific reference discriminant analysis with application in human behavior analysis. *IEEE Transactions on Human-Machine Systems* 45 (2015), 315–326.

[29] A. Iosifidis and M. Gabbouj. On the kernel extreme learning machine speedup. *Pattern Recognition Letters* 68 (2015), 205–210.

[30] A. Iosifidis, A. Tefas and I. Pitas. Approximate kernel extreme learning machine for large scale data classification. *Neurocomputing* 219 (2017), 210–220.

[31] Y. Jia, F. Nie and C. Zhang. Trace ratio problem revisited. *IEEE Transactions on Neural Networks* 20.4 (2009), 729–735.

[32] M. Kan, S. Shan, H. Zhang, S. Lao and X. Chen. Multi-view Discriminant Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38 (2016), 188–194.

[33] T. Kanade, J. Cohn and Y. Tian. Comprehensive database for facial expression analysis. *IEEE International Conference on Automatic Face and Gesture Recognition*. Grenoble, France, (2000), 46–53.

[34] N. Kwak. Implementing kernel methods incrementally by incremental nonlinear projection trick. *IEEE Transactions on Cybernetics* 47 (2017), 4003–4009.

[35] N. Kwak. Nonlinear projection trick in kernel methods: An alternative to the kernel trick. *IEEE Transactions on Neural Networks and Learning Systems* 24.12 (2013), 2113–2119.

[36] K. Lee, J. Ho and D. Kriegman. Acquiring linear subspaces for face recognition under variable lightning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27 (2005), 684–698.

[37] Y. Li, F. Nie, H. Huang and J. Huang. Large-scale multi-view spectral clustering via bipartite graph. *AAAI Conference on Artificial Intelligence*. (2015).

[38] J. Long, E. Shelhamer and T. Darrell. Fully convolutional networks for semantic segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition*. (2015), 3431–3440.

[39] M. Lyons, S. Akamatsu, M. Kamachi and J. Gyoba. Coding facial expressions with Gabor wavelets. *IEEE International Conference on Automatic Face and Gesture Recognition*. Nara, Japan, (1998), 200–205.

[40] B. Ma, H. Qu and H. Wong. Kernel clustering-based discriminant analysis. *Pattern Recognition* 40 (2006), 324–327.

[41] A. Maronidis, A. Tefas and I. Pitas. Graph embedding exploiting subclasses. *IEEE Symposium Series on Computational Intelligence*. Cape Town, South Africa, (2015).

[42] A. Maronidis, A. Tefas and I. Pitas. Subclass graph embedding and a marginal Fisher analysis paradigm. *Pattern Recognition* 48 (2015), 4024–4035.

[43] S. Oramas, F. Barbieri and O. Nieto. Multimodal deep learning for music genre classification. *Transactions of the International Society for Music Information Retrieval* 1 (2018).

[44] S. Pang, S. Ozawa and N. Kasabov. Incremental linear discriminant analysis for classification of data streams. *IEEE Transactions on Systems, Man, and Cybernetics* 35.5 (2005), 905–914.

[45] O. M. Parkhi, A. Vedaldi, A. Zisserman et al. Deep face recognition. *bmvc*. Vol. 1. 3. (2015), 6.

[46] C. R. Rao. The utilization of multiple measurements in problems of biological classification. *Journal of the Royal Statistical Society. Series B (Methodological)* 10.2 (1948), 159–203.

[47] J. Redmon and A. Farhadi. YOLO9000: better, faster, stronger. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. (2017), 7263–7271.

[48] S. Ren, K. He, R. Girshick and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*. (2015), 91–99.

[49] B. Scholkopf., S. Mika, C. Burges, P. Knirsch, K. Muller, G. Ratsch and A. Smola. Input space versus feature space in kernel-based methods. *IEEE Transactions on Neural Networks* (1999), 1000–1017.

[50] B. Scholkopf, A. Smola and K. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation* 10 (1998), 1299–1319.

[51] F. Schroff, D. Kalenichenko and J. Philbin. Facenet: A unified embedding for face recognition and clustering. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. (2015), 815–823.

[52] Y. Shuicheng et al. Graph embedding and extensions: a general framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29 (2007), 40–51.

[53] V. Sigillito, S. Wing, L. Hutton and K. Baker. Classification of radar returns from the ionosphere using neural networks. *Johns Hopkins APL Technical Digest* 10 (1989), 262–266.

[54] J. Smith, J. Everhart, W. Dickson, W. Knowler and R. Johannes. Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. *Proceedings of the Symposium on Computer Applications and Medical Care* (1988), 261–265.

[55] S. Sun. A survey of multi-view machine learning. *Neural Computing and Applications* 23.7-8 (2013), 2031–2038.

[56] S. Torres, D. Ranasinghe, Q. Shi and A. Sample. Sensor enabled wearable RFID technology for mitigating the risk of falls near beds. *IEEE International Conference on RFID*. (2013), 191–198.

[57] H. Wang, X. Lu and W. Zheng. Fisher discriminant analysis with L1-norm. *IEEE Transactions on Cybernetics* 4 (2014), 828–842.

[58] J. Wnek and R. Michalski. Comparing symbolic and subsymbolic learning: three studies. *Machine Learning: A Multistrategy Approach* 4 (1993).

[59] H. Ye, Y. Li, C. Chen and Z. Zhang. Fast Fisher discriminant analysis with randomized algorithms. *Pattern Recognition* 72 (2017), 82–92.

[60] J. Ye. Least squares linear discriminant analysis. *International Conference on Machine Learning* 1 (2007), 1087–1093.

[61] R. A. Yeh et al. Semantic image inpainting with deep generative models. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. (2017), 5485–5493.

[62] L. Yin, X. Wei, Y. Sun, J. Wang and M. Rosato. A 3D facial expression database for facial behavior research. *IEEE International Conference on Automatic Face and Gesture Recognition*. Southampton, UK, (2006), 211–216.

[63] Q. Yin, S. Wu and L. Wang. Unified subspace learning for incomplete and unlabeled multi-view data. *Pattern Recognition* 67 (2017), 313–327.

[64] X. You et al. Multi-view Common Component Discriminant Analysis for Cross-view Classification. *Pattern Recognition* (2019).

[65] C. Zhao et al. Maximal granularity structure and generalized multi-view discriminant analysis for person re-identification. *Pattern Recognition* 79 (2018), 79–96.

[66] M. Zhu and A. Martinez. Subclass discriminant analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28 (2006), 1274–1286.

[67] P. Zhu, Q. Hu, Q. Hu, C. Zhang and Z. Feng. Multi-view label embedding. *Pattern Recognition* 84 (2018), 126–135.