Speed-up and multi-view extensions to subclass discriminant analysis

Kateryna Chumachenko [a,*], Jenni Raitoharju [a,c], Alexandros Iosifidis [b], Moncef Gabbouj [a]

[a] *Faculty of Information Technology and Communication Sciences, Tampere University, Tampere FI-33720, Finland*
[b] *Department of Engineering, Aarhus University, Aarhus DK-8200, Denmark*
[c] *Programme for Environmental Information, Finnish Environment Institute, FI-40500 Jyväskylä, Finland*

ARTICLE INFO

ABSTRACT

In this paper, we propose a speed-up approach for subclass discriminant analysis and formulate a novel efficient multi-view solution to it. The speed-up approach is developed based on graph embedding and spectral regression approaches that involve eigendecomposition of the corresponding Laplacian matrix and regression to its eigenvectors. We show that by exploiting the structure of the between-class Laplacian matrix, the eigendecomposition step can be substituted with a much faster process. Furthermore, we formulate a novel criterion for multi-view subclass discriminant analysis and show that an efficient solution to it can be obtained in a similar manner to the single-view case. We evaluate the proposed methods on nine single-view and nine multi-view datasets and compare them with related existing approaches. Experimental results show that the proposed solutions achieve competitive performance, often outperforming the existing methods. At the same time, they significantly decrease the training time.

1. Introduction

In the modern world, large amounts of data available for training of machine learning algorithms result in their applicability and efficiency in different subject areas [1,2]. However, when the dimensionality of data is high, the algorithms can become susceptible to the well-known curse of dimensionality, stating that in the cases of high-dimensional data, its representation becomes sparse and, therefore, huge amounts of training data are required for the estimation of the parameters of a machine learning method. To address this problem, many dimensionality reduction methods were proposed over the recent years, acquiring an important role within the machine learning field. The objective of the dimensionality reduction methods is to determine a feature space, projection onto which results in a lower dimensionality of data, while preserving properties of the data that are of interest for the problem at hand.

Subspace learning methods can be divided into unsupervised and supervised ones, i.e., those relying solely on the structure of data and those exploiting additional class label information provided by experts. Among the unsupervised dimensionality reduction methods, probably the most common one is Principal Compo-

nent Analysis (PCA) [3], that projects the data onto the subspace where the data has the highest variance.

Supervised subspace learning methods assume that during training the data is given with class labels. Therefore, they lead to enhanced class discrimination compared to unsupervised methods and they are more suitable for classification problems. One of the most well-known methods incorporating the information on class distribution is Linear Discriminant Analysis (LDA) [4,5], where the optimal subspace is obtained by optimizing the Fisher - Rao's criterion [6] that is defined over the within-class and between-class scatter matrices, under the assumption that the classes are unimodal and follow normal distribution. While incorporating the class label information, LDA can only define a subspace of at most d dimensions, where d is the rank of the between-class scatter matrix, which is equal to $C - 1$ for the case of C classes.

The assumption of the class unimodality in LDA limits its performance in problems where classes form subclasses, i.e., classes are represented by multiple disjoint distributions. In order to address this limitation, approaches incorporating the subclass information in the optimization problem solved for determining the discriminant subspace have been proposed. Methods following this approach are the Subclass Discriminant Analysis (SDA) [7], Clustering Discriminant Analysis (CDA) [8], and Subclass Marginal Fisher Analysis (SMFA) [9]. In addition to better describing the classes' distributions, these methods are also able to determine discriminant subspaces of higher dimensionalities, since the maximum dimensionality of the learned feature space is limited by the rank of

* Corresponding author.
E-mail addresses: kateryna.chumachenko@tuni.fi (K. Chumachenko), jenni.raitoharju@tuni.fi (J. Raitoharju), alexandros.iosifidis@eng.au.dk (A. Iosifidis), moncef.gabbouj@tuni.fi (M. Gabbouj).

a modified between-class scatter matrix which is bounded by the total number of subclasses.

One of the main drawbacks of the subspace learning methods lies in the low speed for high-dimensional data and large datasets. For speeding up the training process several approaches have been proposed, including approximate solutions [1], incremental learning [10], and speed-up solutions [11–15]. However, none of these approaches are able to overcome the assumption of unimodality and limitations related to limited dimensionality of the learnt subspace. Therefore, we aim to bridge the speed-up and multi-modal solutions and propose a method that can overcome all of the main limitations of LDA and related methods at once. We achieve this by proposing a speed-up approach for Subclass Discriminant Analysis that already allows to overcome the unimodality assumption and limited potential dimensionality of the learned subspace. In this paper, we propose a speed-up approach for SDA and its kernelized form, i.e., Kernel Subclass Discriminant Analysis (KSDA) [16]. The proposed approach is based on graph embedding [9,17] and exploitation of the structure of the between-class Laplacian matrix.

In some problems, the descriptions of the same items from multiple differently distributed modalities might be available, resulting in multiple modalities of the data. Such problems are referred to as multi-view or multimodal problems. The nature of multi-view problems is similar to the way humans perceive the world and take decisions, as the real-world data is not limited to one source, but consists of, e.g., visual and audio signals, tactile sensations. The data from different modalities is perceived by human and the decision is made by combining information from different sources. A similar approach is followed by multi-view subspace learning methods, where the combination of the information coming from different views is performed by defining a latent feature space, jointly determined using data from all available views during the training process. Moreover, the views can have different dimensionalities. An example of a multi-view problem is the classification of video sequences using their two views, i.e., audio and visual signals.

Extensions of supervised subspace learning methods to the multi-view case include the Multi-view Discriminant Analysis (MDA) [18] that defines a variant of the LDA criterion to incorporate information from multiple views. In [18], the between-class scatter is maximized regardless of the difference between inter-view and intra-view covariances, while the within-class scatter is minimized. Multi-view Common Component Discriminant Analysis proposes a way to address the nonlinearity, view discrepancy and discriminability jointly by incorporating both label information and geometric information during subspace learning [19]. In order to address the problem of multi-label classification with a high number of classes on a multi-view dataset, a Multi-view Label Embedding model was proposed [20]. Besides, for the problems with incomplete or incompletely labeled multi-view data, a unified subspace learning framework has been proposed [21]. In addition, several multi-view extensions of LDA have been recently proposed, including Standard Multi-view Discriminant Analysis (SMvDA) and Multi-view Modular Discriminant Analysis (MvMDA) [22]. Being extensions of LDA, these methods have similar limitations: the assumption of the unimodality of data within each view and maximal number of dimensions bounded by the number of classes. In this work, we propose an approach to overcome these limitations by introducing Multi-view Subclass Discriminant Analysis, as well as its kernelized form, and show that the solution for its optimization problem can be obtained by following a fast and efficient process.

The proposed work brings the following contributions:

- First, we define the Graph Embedding and Spectral Regression -based formulations of Subclass Discriminant Analysis and Kernel Subclass Discriminant Analysis.

- Second, based on the previously-defined formulations, we show how the exploitation of the properties of constant-sum block matrices and the specific structure of the between-class Laplacian matrix can be utilized for speeding up the method. The speed up is achieved by solving the slow eigendecomposition step, which is the main computational bottleneck of SDA, using a fast process of creation of target vectors based on the above-mentioned properties.

- Third, we define a novel multi-view formulation of SDA and KSDA that allows to apply the methods on multi-view data and take into account the potential multi-modalities present in different views. We show how the speed-up approach defined for the single-view formulation can be modified for the multi-view case, resulting in a speed up of the algorithm.

2. Related work

This section describes the previous works related to the proposed supervised subspace learning methods.

Let us consider a set of N D-dimensional vectors $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N] \in \mathbb{R}^D$, each belonging to a class indicated by the corresponding label c_i. We define the subspace learning problem as searching for the d-dimensional feature space, with $d < D$, that provides the highest class separability of the data in $\mathbf{X}$ when projected onto that space. Most dimensionality reduction methods, including LDA, SDA, CDA, and SMFA optimize the Fisher-Rao's criterion [6]:

$$\mathcal{J}(\mathbf{W}) = \underset{\mathbf{W}^T\mathbf{W}=\mathbf{I}}{argmin} \frac{Tr(\mathbf{W}^T\mathbf{S}_w\mathbf{W})}{Tr(\mathbf{W}^T\mathbf{S}_b\mathbf{W})}, \tag{1}$$

where $Tr()$ denotes the trace operator, $\mathbf{S}_w$ and $\mathbf{S}_b$ are symmetric positive semi-definite matrices, referred to as within-class and between-class scatter matrices. The main differences between the subspace learning methods lie in the definition of these matrices. LDA [4] assumes that each class is unimodal and seeks to find a space, projection onto which would result in compact classes lying far from each other, hence, resulting in high discrimination between classes. The within-class and between-class scatter matrices are defined as

$$\mathbf{S}_w = \sum_{i=1}^{C} \sum_{j=1}^{N_i} (\mathbf{x}_{ij} - \boldsymbol{\mu}_i)(\mathbf{x}_{ij} - \boldsymbol{\mu}_i)^T, \tag{2}$$

$$\mathbf{S_b} = \sum_{i=1}^{C} (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T, \tag{3}$$

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 jth sample of class i.

Many extensions to LDA have been proposed over the recent years. Methods relaxing the assumption of LDA about normally distributed classes and the limitations on the dimensionality of the learned subspace in binary problems have been recently proposed in [23–25].

CDA [8] relaxes the assumption on unimodal classes and applies clustering techniques to incorporate the subclass structure of the data in the training process. SMFA relies on a framework of Subclass Graph Embeddings [9], where the dimensionality reduction problem is described from a graph embedding perspective. The problem is defined by intrinsic and penalty graph matrices, which are built relying on the label information of k nearest neighbors of the data points, as defined by the Euclidean distance or some other distance metric. The intrinsic graph matrix represents the compactness within the subclass, while the penalty graph matrix enforces penalization to ensure inter-class separability.

2.1. Graph embedding framework

A framework that considers different subspace learning algorithms from a graph embedding perspective has been proposed in [26]. A further extension to subclass-based methods has been proposed in [9]. In both of these frameworks, data is described using two undirected weighted graphs: an intrinsic graph $\mathbf{G} = \{\mathbf{X}, \mathbf{\Omega}\}$ with vertex set $\mathbf{X}$ and similarity matrix $\mathbf{\Omega}$, and a penalty graph $\mathbf{G}^p = \{\mathbf{X}, \mathbf{\Omega}^p\}$ that represents the similarity characteristics of the data that are desired to be suppressed in the learned space. Each vertex in $\mathbf{X}$ corresponds to a data sample. For each pair of vertices in $\mathbf{X}$, $\mathbf{\Omega}$ measures their similarity by means of some similarity criterion, e.g., Gaussian similarity. Then, the diagonal degree matrix $\mathbf{D}$, the Laplacian matrix $\mathbf{L}$ are defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{\Omega},$$
$$\mathbf{D}^{ii} = \sum_{j \neq i} \mathbf{\Omega}^{ij}, \tag{4}$$

i.e., the degree matrix $\mathbf{D}$ at position (i, i) has the value of the sum of all values of $\mathbf{\Omega}$ across $i'th$ row or column, as $\mathbf{\Omega}$ is symmetric. The penalty Laplacian matrix $\mathbf{L}_b$ can be defined similarly using the penalty matrix $\mathbf{\Omega}_p$. The goal of graph embedding is 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.

Let us define a low-dimensional representation of vertices in $\mathbf{X}$ as $\mathbf{y} = [\mathbf{y}_1, \ldots, \mathbf{y}_i]$. Then, the objective function that solves the problem of finding such a projection matrix $\mathbf{W}$ that would incorporate the similarity between the vertices in $\mathbf{X}$ can be defined as follows:

$$\mathbf{y}^* = argmin \sum_{i \neq j} ||\mathbf{y}_i - \mathbf{y}_j||^2 \mathbf{\Omega}_{ij} = argmin \, \mathbf{y}^T \mathbf{L} \mathbf{y} = argmin \, \mathbf{w}^T \mathbf{X} \mathbf{L} \mathbf{X}^T \mathbf{w}, \tag{5}$$

given $\mathbf{y} = \mathbf{X}^T \mathbf{w}$ is the projection of the data point to a subspace. Similarly, a maximization problem can be defined using $\mathbf{L}_b$. Such formulation allows to reformulate various subspace learning methods and take advantage of the new formulations, as will be shown further.

2.2. Subclass discriminant analysis

In order to relax the class unimodality assumption of LDA, SDA [7] expresses each class by a set of subclasses that are obtained by applying clustering on the class data. The difference between CDA and SDA lies in the definition of the within-class and between-class scatter matrices. In SDA, the total scatter matrix $\mathbf{S}_t$ is minimized instead of the within-class scatter as $\mathbf{S}_t = \mathbf{S}_b + \mathbf{S}_w$. SDA uses the following definitions:

$$\mathbf{S}_t = \sum_{q=1}^{N} (\mathbf{x}_q - \boldsymbol{\mu})(\mathbf{x}_q - \boldsymbol{\mu})^T, \tag{6}$$

$$\mathbf{S}_b = \sum_{i=1}^{C-1} \sum_{l=i+1}^{C} \sum_{j=1}^{d_i} \sum_{h=1}^{d_l} p_{ij} p_{lh} (\boldsymbol{\mu}_{ij} - \boldsymbol{\mu}_{lh})(\boldsymbol{\mu}_{ij} - \boldsymbol{\mu}_{lh})^T, \tag{7}$$

where $\boldsymbol{\mu}$ is the mean of data, i and l are class labels, j and h are subclass labels, p_{ij} and p_{lh} are the subclass priors, $p_{ij} = \frac{N_{ij}}{N}$, where N_{ij} is the number of samples in subclass j of class i and N is the total number of samples in $\mathbf{X}$. The solution of (1) is given by solving the generalized eigendecomposition problem

$$\mathbf{S}_t \mathbf{w} = \lambda \mathbf{S}_b \mathbf{w}. \tag{8}$$

The obtained eigenvectors $[\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_d]$ that correspond to d minimal eigenvalues form a projection matrix $\mathbf{W}$. The projected data point $\mathbf{y}_i$ can be computed as $\mathbf{y}_i = \mathbf{W}^T \mathbf{x}_i$.

It is trivial to see that for the data centered at $\boldsymbol{\mu}$, $\mathbf{S}_t = \mathbf{X}\mathbf{X}^T$. In addition, the representation of $\mathbf{S}_b$ can be defined using the Graph Embedding framework as follows:

$$\mathbf{S}_b = \mathbf{X} \mathbf{L}_b \mathbf{X}^T, \tag{9}$$

$$\mathbf{L}_b(i, j) = \begin{cases} \frac{N - N_{c_i}}{N^2 N_{ch}}, & \text{if } z_i = z_j = h \\ 0, & \text{if } z_i \neq z_j, \ c_i = c_j, \\ -\frac{1}{N^2}, & \text{if } c_i \neq c_j \end{cases} \tag{10}$$

where c_i is the class label of $\mathbf{x}_i$, and z_i is the subclass label of $\mathbf{x}_i$, N_c is the number of samples in class c and N_{ch} is the number of samples in subclass h of class c.

The objective function of SDA can be reformulated into a maximization problem (11), and exploiting the formulations in (9) and (10), the solution is given by the generalized eigendecomposition problem (12), and the projection matrix is obtained by selecting the eigenvectors corresponding to maximal eigenvalues.

$$\mathcal{J}(\mathbf{W}) = \underset{\mathbf{W}^T \mathbf{W} = \mathbf{I}}{argmax} \frac{Tr(\mathbf{W}^T \mathbf{S}_b \mathbf{W})}{Tr(\mathbf{W}^T \mathbf{S}_t \mathbf{W})}, \tag{11}$$

$$\mathbf{L}_b \mathbf{X}^T \mathbf{v} = \lambda \mathbf{X}^T \mathbf{v}. \tag{12}$$

2.3. Kernel subclass discriminant analysis

Kernel methods are widely used in machine learning to overcome the limitation of the linear separability, which is rarely present in real-world problems. In order to nonlinearly map each data point $\mathbf{x}_i$ from the space $\mathbb{R}^D$ to its image $\boldsymbol{\phi}_i$ in some space $\mathcal{F}$, the nonlinear function $\phi(\mathbf{x})$ is defined, i.e., $\phi(\mathbf{x}_i) \in \mathcal{F}$. The dimensionality of $\mathcal{F}$ depends on the choice of the function and can be arbitrary. A linear projection is then defined in $\mathcal{F}$, i.e. $\mathbf{y}_i = \mathbf{W}^T \phi(\mathbf{x}_i)$.

The conventional approach to solving the nonlinear problems involves the exploitation of kernel function defined over a pair of data points in $\mathbf{X}$ that maps them to the dot product of their projections in $\mathcal{F}: k(\mathbf{x}_1, \mathbf{x}_2) = \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_2)$ and formulating the problem accordingly. By exploiting the dot product representation, the explicit mapping of each data point $\mathbf{x}_i$ in $\mathbf{X}$ to its image $\boldsymbol{\phi}_i = \phi(\mathbf{X})$ can be omitted, hence, avoiding the issues related to the arbitrary dimensionality of $\mathcal{F}$. The $N \times N$ kernel matrix $\mathbf{K}$ is defined as $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. It is easy to note 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 $\mathbf{\Phi} = [\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \ldots, \phi(\mathbf{x}_N)]$. According to the Representer Theorem [27], $\mathbf{W}$ can be represented as a linear combination of data in $\mathcal{F}$

$$\mathbf{W} = \mathbf{\Phi} \mathbf{A}. \tag{13}$$

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

The kernelization of the SDA can be easily obtained by exploiting the modified representation of $\mathbf{S}_b$ and $\mathbf{S}_t$ (9) [16]. Here we can assume that data is centered in $\mathcal{F}$. The kernel matrix of the centered data can be obtained as in (14) [28]

$$\mathbf{K}^c = (\mathbf{I} - \mathbf{E}_N) \mathbf{K} (\mathbf{I} - \mathbf{E}_N), \tag{14}$$

$$\mathbf{E}_N = \frac{1}{N} \mathbf{1}_N \mathbf{1}_N^T, \tag{15}$$

where $\mathbf{1}_N \in \mathbb{R}^N$ is a vector of ones.

After mean-centering $\phi(\mathbf{X})$, $\mathbf{S}_{kt}$ and $\mathbf{S}_{kb}$ are given as follows:

$$\mathbf{S}_{kt} = \sum_{i=1}^{N} (\boldsymbol{\phi}_i - \bar{\boldsymbol{\phi}})(\boldsymbol{\phi}_i - \bar{\boldsymbol{\phi}})^T = \boldsymbol{\phi} \boldsymbol{\phi}^T, \tag{16}$$

$$\mathbf{S}_{kb} = \sum_{i=1}^{C-1} \sum_{l=i+1}^{C} \sum_{j=1}^{d_i} \sum_{h=1}^{d_l} p_{ij} p_{lh} (\bar{\boldsymbol{\phi}}_{ij} - \bar{\boldsymbol{\phi}}_{lh})(\bar{\boldsymbol{\phi}}_{ij} - \bar{\boldsymbol{\phi}}_{lh})^T = \boldsymbol{\phi} \mathbf{L}_b \boldsymbol{\phi}^T, \tag{17}$$

where $\bar{\boldsymbol{\phi}}_{ij}$ is the mean of the subclass j of class i in $\mathcal{F}$, $\mathbf{L}_b$ is the between-class Laplacian matrix defined in (10), and $\bar{\boldsymbol{\phi}}$ is the mean of the data in $\mathcal{F}$. Exploiting ((13), (17)-(18)), the solution to KSDA is given by the generalized eigendecomposition problem

$$\boldsymbol{\Phi}\mathbf{L}_b\boldsymbol{\Phi}^T\boldsymbol{\Phi}\mathbf{a} = \lambda\boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{\Phi}\mathbf{a} => \tag{18}$$

$$\mathbf{K}\mathbf{L}_b\mathbf{K}\mathbf{a} = \lambda\mathbf{K}\mathbf{K}\mathbf{a} => \mathbf{L}_b\mathbf{K}\mathbf{a} = \lambda\mathbf{K}\mathbf{a}. \tag{19}$$

2.4. Multi-view extensions to linear discriminant analysis

In multi-view learning, the data $\mathbf{X} = diag(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_V)$ is described from V views and we seek to find V matrices $\mathbf{W}_v$ that project the data $\mathbf{X}_v$ from all views $v = 1, \ldots, V$ to a common (latent) space, where the separability between the classes is the highest. A generalized framework for multi-view subspace learning, that includes many of the existing methods as special cases, was proposed in [22]. Here, the optimization problem is defined as

$$\mathcal{J}(\mathbf{W}) = \underset{\mathbf{W}^T\mathbf{W}=\mathbf{I}}{argmax} \frac{Tr(\mathbf{W}^T\mathbf{P}\mathbf{W})}{Tr(\mathbf{W}^T\mathbf{Q}\mathbf{W})}, \tag{20}$$

where $\mathbf{P}$ and $\mathbf{Q}$ are the inter-view and intra-view covariance matrices. The solution is obtained by solving the generalized eigendecomposition problem

$$\mathbf{P}\mathbf{W} = \rho\mathbf{Q}\mathbf{W}, \tag{21}$$

$$\mathbf{W} = \begin{pmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \\ \ldots \\ \mathbf{W}_V \end{pmatrix}, \tag{22}$$

where $\mathbf{W}_v$ is the projection matrix of the view v. The feature vectors in the latent space are obtained as $\mathbf{Y}_v = \mathbf{W}_v^T\mathbf{X}_v$, where $\mathbf{X}_v$ is data representation in the view v. Here,

$$\mathbf{P} = \mathbf{X}\mathbf{L}_b\mathbf{X}^T, \tag{23}$$

$$\mathbf{Q} = \mathbf{X}\mathbf{L}_w\mathbf{X}^T, \tag{24}$$

$$\mathbf{X} = \begin{pmatrix} \mathbf{X}_1 & 0 & \ldots & 0 \\ 0 & \mathbf{X}_2 & \ldots & 0 \\ 0 & 0 & \ldots & \mathbf{X}_V \end{pmatrix}, \tag{25}$$

$$\mathbf{L}_b = \begin{pmatrix} \mathbf{L}_{b11} & \mathbf{L}_{b12} & \ldots & \mathbf{L}_{bV1} \\ \mathbf{L}_{b12} & \mathbf{L}_{b22} & \ldots & \mathbf{L}_{bV2} \\ \ldots & \ldots & \ldots & \ldots \\ \mathbf{L}_{b1V} & \mathbf{L}_{b2V} & \ldots & \mathbf{L}_{bVV} \end{pmatrix}, \tag{26}$$

$$\mathbf{L}_w = \begin{pmatrix} \mathbf{L}_{w11} & 0 & \ldots & 0 \\ 0 & \mathbf{L}_{w22} & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & \mathbf{L}_{wvv} \end{pmatrix}, \tag{27}$$

where $\mathbf{L}_{bij}$ is either $\mathbf{L}_{bij}^*$ or $\hat{\mathbf{L}}_{bij}$, as defined below, i and j are the view labels, and V is the number of views.

Using the above notations, SMvDA aims to maximize the distance between the class means regardless of the view and defines $\mathbf{L}_{bij}$ as

$$\mathbf{L}_{bij}^* = \begin{cases} 2\sum_{p=1}^{C}\sum_{\substack{q=1 \\ q\neq p}}^{C}(\frac{V}{N_p^2}\mathbf{e}_p\mathbf{e}_p^T - \frac{1}{N_pN_q}\mathbf{e}_p\mathbf{e}_q^T), & \text{if } i = j \\ -2\sum_{p=1}^{C}\sum_{\substack{q=1 \\ q\neq p}}^{C}\frac{1}{N_pN_q}\mathbf{e}_p\mathbf{e}_q^T, & \text{if } i \neq j \end{cases}, \tag{28}$$

where $\mathbf{e}_p$ is N-dimensional class vector with 1s at the positions corresponding to the samples belonging to class p and 0s elsewhere, i and j are views, and C is the number of classes.

The MvMDA maximizes the distances between the centers of different classes across different views:

$$\hat{\mathbf{L}}_{bij} = 2\sum_{p=1}^{C}\sum_{q=1}^{C}(\frac{1}{N_p^2}\mathbf{e}_p\mathbf{e}_p^T - \frac{1}{N_pN_q}\mathbf{e}_p\mathbf{e}_q^T). \tag{29}$$

In both cases, the intra-view Laplacian matrix $\mathbf{L}_w$ is defined as in (27), where

$$\mathbf{L}_{wii} = \mathbf{I} - \sum_{c=1}^{C}\frac{1}{N_c}\mathbf{e}_c\mathbf{e}_c^T, \tag{30}$$

where i is the view label, c is the class label, C is the total number of classes, and $\mathbf{I}$ is the identity matrix. Similarly, the solution to Kernel MvMDA and Kernel SMvDA is given by optimizing

$$\mathcal{J}(\mathbf{A}) = \underset{\mathbf{A}^T\mathbf{K}\mathbf{A}=\mathbf{I}}{argmax} \frac{Tr(\mathbf{A}^T\mathbf{P}^k\mathbf{A})}{Tr(\mathbf{A}^T\mathbf{Q}^k\mathbf{A})}, \tag{31}$$

$$\mathbf{P}^k = \mathbf{K}\mathbf{L}_b\mathbf{K}^T, \tag{32}$$

$$\mathbf{Q}^k = \mathbf{K}\mathbf{L}_w\mathbf{K}^T, \tag{33}$$

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 vth block. The solution is then given by solving the eigendecomposition problem

$$\mathbf{P}^k\mathbf{A} = \rho\mathbf{Q}^k\mathbf{A}. \tag{34}$$

2.5. Spectral regression

In this section, we focus on the spectral regression approach that was introduced as a way of speeding up the eigendecomposition step of LDA [29]. It has been shown that the solution of the generalized eigendecomposition problem (12) is equivalent to the problem $\mathbf{J}\mathbf{t} = \lambda\mathbf{t}$ with the same eigenpairs, for $\mathbf{t} = \mathbf{X}^T\mathbf{w}$ and $\mathbf{J} = \mathbf{L}_b$:

$$\mathbf{J}\mathbf{t} = \mathbf{J}\mathbf{X}^T\mathbf{w} = \lambda\mathbf{X}^T\mathbf{w} = \lambda\mathbf{t}. \tag{35}$$

Exploiting this fact, the solution of (12) can be obtained by solving an eigenvalue decomposition problem $\mathbf{J}\mathbf{t} = \lambda\mathbf{t}$ and finding such $\mathbf{w}$ that $\mathbf{X}^T\mathbf{w} = \mathbf{t}$. In practice, such $\mathbf{w}$ may not always exist, but it can be approximated with the closest value in the least squares sense:

$$\mathbf{W} = argmin\|\mathbf{W}^T\mathbf{X} - \mathbf{T}\|_F^2. \tag{36}$$

The solution to (36) is given by $\mathbf{W} = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{T}^T$. In the cases where $\mathbf{X}\mathbf{X}^T$ is singular, regularized solution is applied:

$$(\mathbf{X}\mathbf{X}^T + \alpha\mathbf{I})\mathbf{W} = \mathbf{X}\mathbf{T}^T, \tag{37}$$

$$\mathbf{W} = (\mathbf{X}\mathbf{X}^T + \alpha\mathbf{I})^{-1}\mathbf{X}\mathbf{T}^T, \tag{38}$$

where $\alpha \geq 0$ is a regularization parameter and $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_d]^T$.

Spectral Regression Discriminant Analysis (SRDA) was proposed as an extension to LDA based on the spectral regression [29]. It has been shown that in the case of LDA the matrix $\mathbf{J}$ (35) has C eigenvectors corresponding to nonzero values, all of which correspond to the eigenvalue of 1 and have the form of

$$\mathbf{u}_i = [\underbrace{0, \ldots, 0}_{\sum_{i=1}^{p-1} N_i}, \underbrace{1, \ldots, 1}_{N_p}, \underbrace{0, \ldots, 0}_{\sum_{i=p+1}^{C} N_i}]^T, \tag{39}$$

where p is the class label, N_p is the number of samples in class p and C is the number of classes. Therefore, the solution can be obtained by selecting the vector of ones as the first eigenvector and

obtaining the rest by orthogonalization of the vectors of the structure as in (39). A tensor extension to SRDA has been recently proposed in [30], where the eigendecomposition problem of Higher Order Discriminant Analysis is transformed into a regression problem.

2.6. Kernel regression

A kernelized version of the spectral regression was proposed in [11]. In this case, the objective is to solve the eigendecomposition problem $\mathbf{JKa} = \lambda \mathbf{Ka}$, which is equivalent to solving the eigendecomposition problem of $\mathbf{Jt} = \lambda \mathbf{t}$ given $\mathbf{Ka} = \mathbf{t}$:

$$\mathbf{JKa} = \mathbf{Jt} = \lambda \mathbf{t} = \lambda \mathbf{Ka}. \tag{40}$$

Then the kernel regression is applied to obtain

$$\mathbf{W}^* = \underset{\mathbf{W}}{argmin}\ ||\mathbf{W}^T \mathbf{\Phi} - \mathbf{T}||_F^2, \tag{41}$$

$$\mathbf{A} = argmin\|\mathbf{A}^T \mathbf{\Phi}^T \mathbf{\Phi} - \mathbf{T}\|_F^2 = argmin\|\mathbf{A}^T \mathbf{K} - \mathbf{T}\|_F^2. \tag{42}$$

To take into account possible singularity of $\mathbf{KK}^T$, regularized solution is used to obtain $\mathbf{A}$:

$$\mathbf{A} = (\mathbf{KK}^T + \alpha \mathbf{I})^{-1} \mathbf{KT}^T, \tag{43}$$

where α is the regularization parameter.

2.7. Approximate kernel regression

For large-scale datasets, kernel regression method can be substituted by an approximate kernel regression, where $\mathbf{W}$ is expressed as a linear combination of r reference vectors ($r < N$) [1]. We define $\mathbf{W} = \mathbf{\Psi A}$, where $\mathbf{\Psi}$ is a set of reference vectors in $\mathcal{F}$. The reference vectors in $\mathcal{F}$ correspond to r prototype vectors from $\mathbb{R}^D$ that can be randomly selected training vectors from $\mathbf{X}$, random data following the same distribution as data in $\mathbf{X}$, subclass centers obtained by clustering all data, or subclass centers obtained by clustering data in each subclass separately.

Given $\mathbf{W} = \mathbf{\Psi A}$, (42) becomes

$$\mathbf{A}^* = argmin\|\mathbf{A}^T \mathbf{\Psi}^T \mathbf{\Phi} - \mathbf{T}\|_F^2 = argmin\|\mathbf{A}^T \hat{\mathbf{K}} - \mathbf{T}\|_F^2, \tag{44}$$

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

$$\mathbf{A} = (\hat{\mathbf{K}}\hat{\mathbf{K}}^T + \alpha \mathbf{I})^{-1} \hat{\mathbf{K}} \mathbf{T}^T, \tag{45}$$

where α is a regularization parameter. It should be noted that in the case $\mathbf{\Psi} = \mathbf{\Phi}$, the problem becomes equivalent to (43).

2.8. SDA with spectral regression

Subclass Discriminant Analysis has not been previously used together with Spectral Regression, but their combination is straightforward. The process of solving SDA using Spectral Regression can be defined as follows:

1. Create the between-class Laplacian matrix (10)
2. Solve the generalized eigendecomposition problem $\mathbf{L}_b \mathbf{t} = \lambda \mathbf{t}$ and create the matrix $\mathbf{T}$ out of the obtained vectors
3. Regress $\mathbf{T}$ to $\mathbf{W}$ as in (38)
4. Orthogonalize $\mathbf{W}$ such that $\mathbf{W}^T \mathbf{W} = \mathbf{I}$

Equivalently, for the kernel case, the steps 3–4 are the regression of $\mathbf{T}$ to $\mathbf{A}$ as in (43) or (45) and orthogonalization of $\mathbf{A}$ such that $\mathbf{A}^T \mathbf{KA} = \mathbf{I}$. Alternatively, the projection matrix can be l2-normalized instead of applying orthogonalization.

The above-described process for solving the SDA optimization problem provides several advantages. Firstly, as we will show in the next section, the eigendecomposition step (35) can be substituted with a much faster process. Secondly, the eigendecomposition step (12) or (19) is avoided and substituted with the least squares regression, for which several efficient solutions exist [31].

3. Proposed approach

In this section, the proposed methods are described. Firstly, we propose a speed-up approach for single-view SDA that relies on the structure of the Laplacian matrix $\mathbf{L}_b$ and allows to substitute the eigendecomposition step of (35) by a much faster process. Secondly, we propose a linear and kernel solutions for multi-view SDA. Thirdly, we show that the solution to multi-view SDA can be obtained by a faster process that is similar to the one described for the single-view case.

3.1. Speeding up the eigendecomposition step

In this section, we show how the specific block structure of the Laplacian matrix $\mathbf{L}_b$ in SDA allows to replace the eigendecomposition step with a much faster process.

Without loss of generality, we assume that the data in $\mathbf{X}$ is mean-centered and sorted according to the class and subclass labels, i.e., $[1, \ldots, N_{11}, 1, \ldots, N_{CZ}]$, where $[1, \ldots, N_{CZ}]$ are the subclass labels of class C and subclass Z.

It can be observed that $\mathbf{L}_b$ has a block structure with constant values in the blocks, as described in (10), with different blocks of $\mathbf{L}_b$ corresponding to different classes. The class blocks are further divided into the subclass blocks. Since $\mathbf{L}_b$ has a block structure, its eigenvectors have the block structure as well. Moreover, bigger eigenvalues show larger differentiation and correspond to the eigenvectors discriminating class blocks, while smaller eigenvalues discriminate subblocks of class blocks, hence, representing subclasses. $\mathbf{L}_b$ has a rank of $C * Z - 1$ and, therefore, it has $C * Z - 1$ nonzero eigenvalues, where Z is the number of subclasses in each class.

Assuming the eigenvectors are sorted according to the eigenvalues in decreasing order, the first $C - 1$ eigenvectors share similar values at indices corresponding to one class. The rest of the eigenvectors correspond to different classes, and in each of them the subclass structure of a certain class can be observed - the indices corresponding to data of the same subclass have the same nonzero value, while the indices corresponding to other classes have the value of 0. We observe that bigger eigenvalues correspond to the eigenvectors showing the subclass discrimination of classes with smaller number of samples; and the classes having the same amount of samples share the eigenvectors, i.e., samples at positions of both classes have nonzero values, that are the same within a subclass, while positions corresponding to other classes have the value of zero. In this case, such eigenvectors are repeated a number of times equal to the number of classes with the same amount of samples.

As an example, let us consider a problem of 2 classes, where class 1 contains 8 samples and class 2–9 samples. Each class contains 2 subclasses, where class 1 has 3 samples in the first subclass and 5 in the second, and class 2 has 4 samples in the first subclass and 5 samples in the second subclass. Then the three eigenvectors of the between-class Laplacian matrix of this data that correspond to nonzero eigenvalues have the structure outlined in (46), where c corresponds to the class label, z - to subclass label and r_i - to ith random value.

Moreover, $\mathbf{L}_b$ is a symmetric weightless constant sum matrix. Therefore, all of its eigenvectors are orthogonal and a vector of ones is an eigenvector with eigenvalue 0 [32]. In addition, we can observe that for the data with a subclass structure, the eigenvectors maximizing the criterion (11) are those with the block structure as described. Following this, the orthogonalization can be performed on random vectors that follow the block structure as described above [33]. Therefore, we can choose the vector of ones as our first eigenvector and obtain the remaining $C * Z - 1$ vectors by orthogonalizing the random vectors of the described structure

Algorithm 1: Target vectors calculation, single-view case.

Function

getSingleviewTargets(*class_labels*, *cluster_labels*, *C*, *Z*, *N*, *D*):

> **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; *D* : dimensionality of data;
>
> $d \leftarrow \min(C*Z-1, D, N)$;
>
> %class-level vectors;
>
> $T \leftarrow N \times (\min(d, C-1))$ 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**
>> $k \leftarrow$ classes with l elements; $m \leftarrow$ length(k);
>> **if** $size(T, 2) + m*(Z-1) > d$ **then**
>>> $m \leftarrow (d - size(T, 2))/(Z-1)$
>>
>> **end**
>> $Tclust \leftarrow N \times m*(Z-1)$ 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;
>> **if** $size(T, 2) == d$ **then**
>>> **break**
>>
>> **end**
>
> **end**
>
> $T \leftarrow$ append $N \times 1$ vector of ones as a column on the left;
> Orthogonalize T; remove first column of T;
> **return** T^T

---

following the Gram-Schmidt process [34]. As $d \leq \min(D, N)$, in the case $C*Z-1 > \min(D, N)$ we can stop after $min(D, N)$ target vectors are created. The vector of ones can then be removed as being useless. The detailed process of target vectors creation is outlined in Algorithm 1.

$$
\begin{pmatrix}
r_1 & r_3 & 0 \\
r_1 & r_3 & 0 \\
r_1 & r_3 & 0 \\
r_1 & r_4 & 0 \\
r_1 & r_4 & 0 \\
r_1 & r_4 & 0 \\
r_1 & r_4 & 0 \\
r_1 & r_4 & 0 \\
r_2 & 0 & r_5 \\
r_2 & 0 & r_5 \\
r_2 & 0 & r_5 \\
r_2 & 0 & r_5 \\
r_2 & 0 & r_6 \\
r_2 & 0 & r_6 \\
r_2 & 0 & r_6 \\
r_2 & 0 & r_6 \\
r_2 & 0 & r_6
\end{pmatrix}
\begin{matrix}
c=1, z=1 \\
c=1, z=1 \\
c=1, z=1 \\
c=1, z=2 \\
c=1, z=2 \\
c=1, z=2 \\
c=1, z=2 \\
c=1, z=2 \\
c=2, z=1 \\
c=2, z=1 \\
c=2, z=1 \\
c=2, z=1 \\
c=2, z=2 \\
c=2, z=2 \\
c=2, z=2 \\
c=2, z=2 \\
c=2, z=2
\end{matrix} .
\tag{46}
$$

3.2. Multi-view subclass discriminant analysis

In this section, we propose a novel method for multi-view subspace learning - Multi-view Subclass Discriminant Analysis along with the kernelized version. Unlike previously described SMvDA and MvMDA methods that similarly to LDA assume that data within each view follows a unimodal Gaussian distribution, we propose a method that would take into account potential multimodalities present in the data of each view and hence lead to a more robust solution. This is done by modelling data of each view with multiple subclasses. Therefore, the idea behind multi-view Subclass Discriminant Analysis is the maximization of the distance between the subclass means of different classes, while minimizing the distances between the samples of the same subclass. The total scatter matrix for the mean-centered data is defined as

$$
\mathbf{S}_t = \sum_{i=1}^{V} \sum_{k=1}^{N} \mathbf{y}_k^i \mathbf{y}_k^{i\,T} = \mathbf{YY}^T = \mathbf{W}^T \mathbf{XX}^T \mathbf{W},
\tag{47}
$$

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

$$
\mathbf{S}_b = \sum_{i=1}^{V} \sum_{j=1}^{V} \sum_{p=1}^{C} \sum_{\substack{q=1 \\ q \neq p}}^{C} \sum_{l=1}^{d_p} \sum_{h=1}^{d_q} p_{pl}^i p_{qh}^j (\boldsymbol{\mu}_{pl}^i - \boldsymbol{\mu}_{qh}^j)(\boldsymbol{\mu}_{pl}^i - \boldsymbol{\mu}_{qh}^j)^T
$$

$$
= \sum_{i=1}^{V} \sum_{j=1}^{V} \mathbf{W}_i^T \mathbf{X}_i \mathbf{L}_{bij}^{mv} \mathbf{X}_j^T \mathbf{W}_j = \mathbf{W}^T \mathbf{X} \mathbf{L}_b^{mv} \mathbf{X}^T \mathbf{W},
\tag{48}
$$

$$
\mathbf{X} = \begin{pmatrix} \mathbf{X}_1 & 0 & \dots & 0 \\ 0 & \mathbf{X}_2 & \dots & 0 \\ 0 & 0 & \dots & \mathbf{X}_V \end{pmatrix},
\tag{49}
$$

$$
\mathbf{W} = \begin{pmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \\ \dots \\ \mathbf{W}_V \end{pmatrix},
\tag{50}
$$

$$
\mathbf{L}_b^{mv} = \begin{pmatrix} \mathbf{L}_{b11}^{mv} & \mathbf{L}_{b21}^{mv} & \dots & \mathbf{L}_{bV1}^{mv} \\ \mathbf{L}_{b12}^{mv} & \mathbf{L}_{b22}^{mv} & \dots & \mathbf{L}_{bV2}^{mv} \\ \dots & \dots & \dots & \dots \\ \mathbf{L}_{b1V}^{mv} & \mathbf{L}_{b2V}^{mv} & \dots & \mathbf{L}_{bVV}^{mv} \end{pmatrix},
\tag{51}
$$

$$
\mathbf{L}_{bij}^{mv} = \begin{cases} 2 \sum_{p=1}^{C} \sum_{\substack{q=1 \\ q \neq p}}^{C} \sum_{l=1}^{d_p} \sum_{h=1}^{d_q} \frac{VN_{qh}^j}{N_{pl}^i N^2} \mathbf{e}_{pl}^i \mathbf{e}_{pl}^{i\,T} - \frac{1}{N^2} \mathbf{e}_{pl}^i \mathbf{e}_{qh}^{j\,T}, & \text{if } i = j \\ \\ -2 \sum_{p=1}^{C} \sum_{\substack{q=1 \\ q \neq p}}^{C} \sum_{l=1}^{d_p} \sum_{h=1}^{d_q} \frac{1}{N^2} \mathbf{e}_{pl}^i \mathbf{e}_{qh}^{j\,T}, & \text{otherwise} \end{cases},
\tag{52}
$$

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 solution is then obtained by optimizing the Fisher-Rao's criterion:

$$
\mathcal{J}(\mathbf{W}) = \underset{\mathbf{W}_i^T \mathbf{W}_i = \mathbf{I}, i=1,\dots,V}{argmax} \frac{Tr(\mathbf{W}^T \mathbf{X} \mathbf{L}_b \mathbf{X}^T \mathbf{W})}{Tr(\mathbf{W}^T \mathbf{XX}^T \mathbf{W})},
\tag{53}
$$

where $\mathbf{X}$ and $\mathbf{W}$ are defined as in (49) and (50), respectively, and $\mathbf{K}$ is centered. Equivalently, solution to the kernel version of the method is obtained by optimizing

$$
\mathcal{J}(\mathbf{A}) = \underset{\mathbf{A}^T \mathbf{KA} = \mathbf{I}}{argmax} \frac{Tr(\mathbf{A}^T \mathbf{K} \mathbf{L}_b \mathbf{K}^T \mathbf{A})}{Tr(\mathbf{A}^T \mathbf{KK}^T \mathbf{A})},
\tag{54}
$$

where $\mathbf{K}$ is a block-diagonal matrix having $\mathbf{K}_v$ as its vth block.

The solution to (53) 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 (54) is given by $\mathbf{L}_b\mathbf{K}\mathbf{a} = \lambda\mathbf{K}\mathbf{a}$. Both of these problems can be solved by the process equivalent to the one described in 3.1.

3.3. Speeding up the eigendecomposition step: multi-view case

In this section, we describe a speed-up approach for the Multi-view Subclass Discriminant Analysis, based on the specific structure of the Laplacian matrix $\mathbf{L}_b^{mv}$. The process of speeding up the eigendecomposition step for the multi-view case is similar to the single-view one. The Laplacian matrix $\mathbf{L}_b^{mv}$ is the constant sum symmetric block matrix, thus having orthogonal eigenvectors, one of which is the vector of ones corresponding to eigenvalue of 0. The matrix has a block structure, where different blocks correspond to different views, and inside of each diagonal view block we can observe the block structure that is the same as in the single-view case. Due to this block structure the eigenvectors of $\mathbf{L}_b^{mv}$ have the block structure as well. Assuming that the number of clusters is the same in all views, the rank of the $\mathbf{L}_b^{mv}$ is $C*Z*V-1$, and that is the maximum number of nonzero eigenvalues.

Let us consider the data of 2 views and 2 classes. Let 1 contain 2 subclasses, with 2 samples in the first subclass and 2 samples in the second subclass in both views. Let class 2 contain 2 subclasses, with 3 samples in the first subclass and 4 samples in the second subclass in the first view, and 4 samples in the first subclass and 3 samples in the second subclass in the second view. Then the eigenvectors of the between-class Laplacian matrix of the proposed multi-view SDA will have the structure outlined in (55), where c corresponds to the class label, z corresponds to the subclass label, v corresponds to the view label and r_i corresponds to ith random value.

$$
\begin{pmatrix}
r_1 & r_5 & r_9 & r_{13} & 0 & 0 & 0 \\
r_1 & r_5 & r_9 & r_{13} & 0 & 0 & 0 \\
r_1 & r_6 & r_{10} & r_{14} & 0 & 0 & 0 \\
r_1 & r_6 & r_{10} & r_{14} & 0 & 0 & 0 \\
r_2 & 0 & 0 & 0 & r_{17} & r_{21} & r_{25} \\
r_2 & 0 & 0 & 0 & r_{17} & r_{21} & r_{25} \\
r_2 & 0 & 0 & 0 & r_{17} & r_{21} & r_{25} \\
r_2 & 0 & 0 & 0 & r_{18} & r_{22} & r_{26} \\
r_2 & 0 & 0 & 0 & r_{18} & r_{22} & r_{26} \\
r_2 & 0 & 0 & 0 & r_{18} & r_{22} & r_{26} \\
r_2 & 0 & 0 & 0 & r_{18} & r_{22} & r_{26} \\
r_3 & r_7 & r_{11} & r_{15} & 0 & 0 & 0 \\
r_3 & r_7 & r_{11} & r_{15} & 0 & 0 & 0 \\
r_3 & r_8 & r_{12} & r_{16} & 0 & 0 & 0 \\
r_3 & r_8 & r_{12} & r_{16} & 0 & 0 & 0 \\
r_4 & 0 & 0 & 0 & r_{19} & r_{23} & r_{27} \\
r_4 & 0 & 0 & 0 & r_{19} & r_{23} & r_{27} \\
r_4 & 0 & 0 & 0 & r_{19} & r_{23} & r_{27} \\
r_4 & 0 & 0 & 0 & r_{19} & r_{23} & r_{27} \\
r_4 & 0 & 0 & 0 & r_{20} & r_{24} & r_{28} \\
r_4 & 0 & 0 & 0 & r_{20} & r_{24} & r_{28} \\
r_4 & 0 & 0 & 0 & r_{20} & r_{24} & r_{28}
\end{pmatrix}
\begin{matrix}
c & z & v \\
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 2 & 1 \\
1 & 2 & 1 \\
2 & 1 & 1 \\
2 & 1 & 1 \\
2 & 1 & 1 \\
2 & 2 & 1 \\
2 & 2 & 1 \\
2 & 2 & 1 \\
2 & 2 & 1 \\
1 & 1 & 2 \\
1 & 1 & 2 \\
1 & 2 & 2 \\
1 & 2 & 2 \\
2 & 1 & 2 \\
2 & 1 & 2 \\
2 & 1 & 2 \\
2 & 1 & 2 \\
2 & 2 & 2 \\
2 & 2 & 2 \\
2 & 2 & 2
\end{matrix}. \quad (55)
$$

It can be observed that the first $C-1$ eigenvectors have the class block structure similar to the one in the single-view case, and the blocks are repeated across the positions corresponding to the different views. In the same way as in the previously described single-view case, the rest of the eigenvectors correspond to different classes and each of them exposes the subclass structure of specific class - the values corresponding to the same subclass are the same within each view in the eigenvector and the values corresponding to other classes are 0 in all the views. We observe that

---

Algorithm 2: Target vectors calculation, multi-view case.

Function
getMultiviewTargets(*class_labels*, *cluster_labels*, V, C, Z, N, D):

  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; D : vector of dimensionalities of data in each view;

  $d \leftarrow \min(V*C*Z-1, \min(D), N)$;

  %class-level vectors;

  $T \leftarrow V*N \times (\min(d, C-1))$ 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**
    $k \leftarrow$ classes with l elements; $m \leftarrow$ length(k);
    if $size(T, 2) + m*(V*Z-1) > d$ **then**
      $m \leftarrow (d - size(T, 2))/(V*Z-1)$
    end
    $Tclust \leftarrow V*N \times m*(V*Z-1)$ 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;
    if $size(T, 2) == d$ **then**
      break
    end
  end

  $T \leftarrow$ append $N \times 1$ vector of ones as a column on the left;
  Orthogonalize T; remove first column of T;
  return T^T

---

the classes with the same amount of samples share the eigenvectors in a similar way to the single-view case, and these eigenvectors are repeated for the number of times equal to the number of classes sharing the number of elements. The eigenvectors showing subclass discrimination of smaller classes correspond to bigger eigenvalues.

Following the procedure described for single-view case, the eigenvectors can be obtained by forming the random vectors of the structure described, and orthogonalizing starting from the vector of ones following the Gram-Schmidt process [34]. The vector of ones can then be removed. The detailed procedure is described in Algorithm 2.

3.4. Computational complexity analysis

In this section, we discuss the complexity analysis of the original Subclass Discriminant Analysis and the proposed speed-up approach. The complexities are described using *flam* - a compound operation denoting one addition and one multiplication [35]. Complexity of SDA can be defined as follows:

- Calculation of total scatter matrix $\mathbf{S}_t$ is $\frac{ND^2}{2} + DN$

- Calculation of between-class scatter matrix $\mathbf{S}_b$ is $\frac{Z^2 C(C-1)}{2}\left(\frac{D(D+1)}{2}+1+D\right)+2DN$, where Z is the number of subclasses and C is a number of classes
- Solving the eigendecomposition problem in (8) is $\frac{9}{2}D^3 + 2D^3 + D^3 = \frac{15}{2}D^3$ [29,35]

It should be noted here that we do not perform the calculation of the stability criterion defined in [7] as the number of subclasses is defined explicitly for fair comparison with other methods - fixing the same cluster labels for data points in all the methods ensures that clustering accuracy plays no effect on overall accuracy and fine-tuning the number of clusters does not affect the speed. Besides, implementation following the SDA_{stab} algorithm requires multiple iterations with different subclass numbers therefore making the algorithm more computationally intensive.

The computational complexity of fastSDA depends on three steps: mean-centering of data, creation of target vectors, and regression step. Let us first consider the regression problem in (38). First, we can notice that for a positive definite matrix $(\mathbf{XX}^T + \alpha\mathbf{I})$, inversion can be performed efficiently via Cholesky decomposition, i.e., $(\mathbf{XX}^T + \alpha\mathbf{I})^{-1} = (\mathbf{R}^{-1})(\mathbf{R}^{-1})^T$, where $\mathbf{R}$ is an upper triangular matrix, s.t., $(\mathbf{XX}^T + \alpha\mathbf{I}) = \mathbf{R}^T\mathbf{R}$. Further, when $N < D$, the problem in (38) can be transformed into $\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{T}^T$. Thus, the complexity of fastSDA can be calculated as follows:

- Complexity of target vector creation is essentially equal to complexity of Gram-Schmidt process $= Nd^2 - \frac{1}{3}d^3$, where $d = ZC - 1$ is the dimensionality of the projection space (number of target vectors), $d < D$ [29,35].
- Complexity of mean-centering data is DN
- If $N \leq D$, complexity of the regression step is $\frac{DN^2}{2} + \frac{N^3}{6} + dN^2 + DNd$, where $\frac{DN^2}{2}$ is the complexity of $\mathbf{X}^T\mathbf{X}$, $\frac{N^3}{6}$ is the complexity of Cholesky decomposition [29,35], and $d^2N + DNd$ is the complexity of the remaining regression steps (i.e., calculation of inverse and multiplication).
- If $N > D$, complexity is $\frac{ND^2}{2} + \frac{D^3}{6} + dD^2 + DNd$, where $\frac{ND^2}{2}$ is the complexity of $\mathbf{XX}^T$, $\frac{D^3}{6}$ is the complexity of Cholesky decomposition [29,35], and $d^2D + DNd$ is the complexity of the remaining regression steps (i.e., calculation of inverse and multiplication).
- Normalization of $\mathbf{W}$ has complexity of $\frac{d^2D}{2} + dD + d$

In the case $N < D$, the most computationally intensive term of fastSDA depends on relation of d to N. It is equal to $\frac{DN^2}{2}$ for $N > 2d$ and DNd for $N < 2d$. Both of these terms are smaller than D^3 as both N and d are smaller than D. In turn, the computational com-

plexity of SDA is at least $\frac{15}{2}D^3$. Therefore, fastSDA will always outperform SDA in this scenario. The dependency of speed-up on dimensionality and the number of samples can be seen from Fig. 1. Here we show the ratio of SDA training time to fastSDA training time for four different cases of D, N, C, Z, where we consider cases of large and small number of classes/subclasses for larger/smaller D. As can be observed, the speed up ratio increases with the increase of D.

In the case $N > D$, the highest term of SDA is at least $\frac{ND^2}{2}$ if $N > 15D$ and $\frac{15}{2}D^3$ otherwise. Besides, the complexity term of calculation of between-class scatter becomes significant for higher C and Z. For fastSDA the largest term is $\frac{ND^2}{2}$ if $d < \frac{D}{2}$ and DNd otherwise. Thus, the speed-up ratio depends on the five parameters of D, N, C, Z, d. The dependency of training time on C and Z is shown on Fig. 2 where four cases are considered: two for a small dataset and two for a large dataset, where in each case we show the subcases where $N > 15D$ and $N < 15D$ and the ratio of training time of SDA to that of fastSDA is shown. It can be observed that for big enough D fastSDA always results in a speed up that becomes more significant with higher D, C, and Z. For the case where D is small, $D << N$, and both C and Z are small, the speed-up might not always be achieved or be close to 1, as can be observed from the top-left subplot. However, we argue that scenario where $D << N$ is not a common case where dimensionality reduction is applied in the first place. Experimentally we also show that our methods results in superior speed for both large-scale datasets with many classes (e.g., SoF) and small low-dimensional datasets with lower number of classes (e.g., Ionosphere, Monks2).

In the kernel case, the regression step (43) is equivalent to $(\mathbf{K} + \alpha\mathbf{I})^{-1}\mathbf{T}^T$ for normalized $\mathbf{A}$. The complexity of kernel fastSDA can then be calculated as follows:

- Complexity of kernel matrix calculation is DN^2
- Complexity of mean-centering of $\mathbf{K}$ is $\frac{N^2}{2} + \frac{N^3}{2}$
- Complexity of target vectors creation is $Nd^2 - \frac{1}{3}d^3$
- Complexity of regression step is $\frac{N^3}{6} + dN^2$
- Complexity of normalization of the projection matrix is $dN^2 + d^2N + dN + d$

For kernel SDA, the step of kernel matrix creation is added as well. Besides, as kernel SDA formulation is based on graph embedding framework, the complexity of KSDA becomes at least equal to the complexity of eigendecomposition (19) which is equal to $\frac{15}{2}N^3$, while the largest term of fastSDA complexity becomes $\frac{2}{3}N^3$ (here we exclude the kernel matrix creation term as it is equal for both methods). The complexity of multi-view fastSDA can be computed
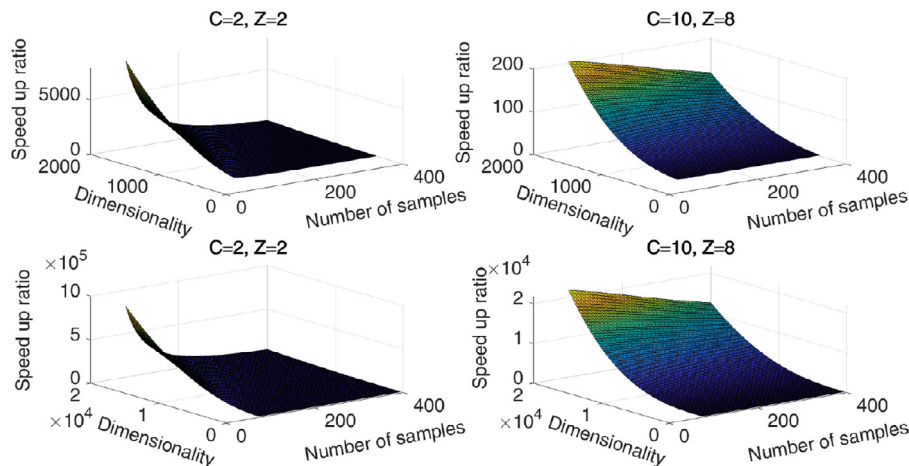

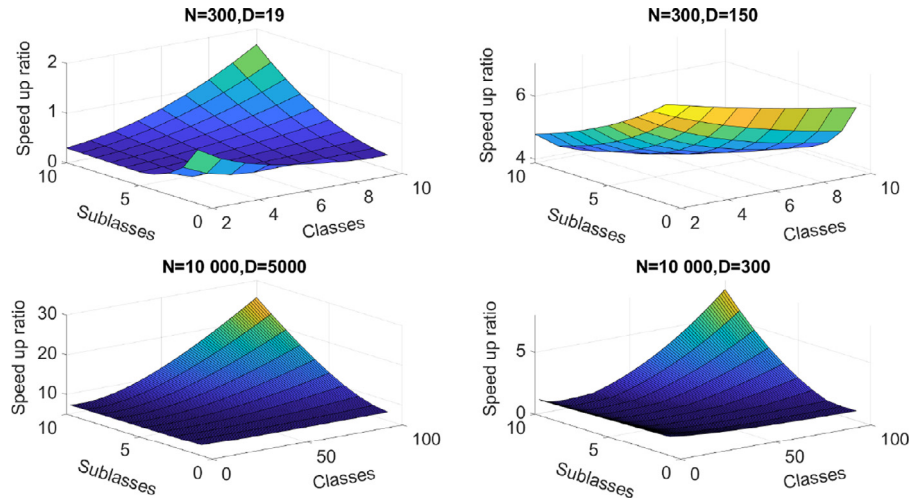**Fig. 1.** Dependancy of speed up ratio based on *N, D, C, Z* for *N < D*.

Fig. 2. Dependancy of speed up ratio based on *N, D, C, Z* for $N > D$.

similarly, following $N = \sum_{i=1}^{V} N_i$ and $D = \sum_{i=1}^{V} D_i$ ($D = \sum_{i=1}^{V} N_i$ in the kernel case), where V is the number of views.

We can conclude that fastSDA outperforms SDA in terms of computational complexity and the speed-up increases with the dimensionality of data. Besides, another speed-up factor comes from the computation of the between-class scatter matrix that becomes much more computationally intensive with larger number of classes and/or subclasses in SDA.

4. Experimental results

In this section, the experimental results are presented. The results are compared with other subspace learning techniques, namely SDA, CDA, SMFA, and SRDA, as well as the kernel SDA, CDA, and SMFA. After feature extraction, classification is performed with k-Nearest Neighbors classifier with $k = 5$. In addition, we verify some of the assumptions regarding the proposed approach by performing eigendecomposition of L_b, regressing the obtained eigenvectors following (38) and projecting the data onto the obtained vectors that correspond to larger criterion values (11).

For the kernel version of the methods, we exploit the RBF kernel function:

$$\mathbf{K}(x_i, x_j) = exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||_2^2}{2\sigma^2}\right), \tag{56}$$

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

In our experiments, we assume that the subclass label of each data point in each class is known and is determined by applying the k-means clustering in $\mathbb{R}^D$. The performance is tested for the different numbers of clusters $Z = \{1, 2, 3, 4, 5, 6\}$, and the same number of clusters is used for each class. We perform clustering in the original space and use the same cluster labels in the kernel methods. The same subclass labels are used for all subclass-based methods to guarantee that the differences in performance observed between the methods are not related to the specific clustering solutions of K-Means, but on the optimization problem each method adopts for determining the corresponding subspace. In the multi-view case, data in each view is clustered separately. The dimensionality d of the projection space is defined by the rank of the $\mathbf{L}_b$ or $\mathbf{L}_b^{mv}$ matrix and is 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 clusters.

For each experiment, 5-fold stratified cross-validation was used, with 60% of data of each class belonging to training set, 20% to

validation set, and 20% to test set, where the validation set is used for hyperparameter tuning, and results are reported by training on the training set and testing on the test set. All experiments were performed on a computer with 4-core Intel i7-4800Q CPU and 32 GB of RAM.

For single-view approaches, prior to using any method, we applied PCA, preserving the eigenvectors corresponding to 98% of the total energy and the data was standardized. The hyperparameters of all methods, if any, were tuned with the grid search. For SMFA, k_{Int} and k_{Pen} were selected from the range of [2,14] with step 3 and [20,100] with step 20, respectively.

For calculating the distance matrix in SMFA/KSMFA, Gaussian similarity (56) with σ equal to the mean Euclidean distance between the training vectors was used. The regularization parameter for the kernel regression was chosen from the set $[1e^{-3}, 1e^{-2}, \ldots, 1e^3]$. For the regularization of the other single-view kernel methods and multi-view methods the same parameter range was used. Cholesky decomposition was used for efficient matrix inversion.

In the multi-view kernel case, the solutions for the datasets containing more than 2500 samples were obtained with approximate kernel regression with the kernel matrix formed with 1500 random vectors from the training data. In the single-view kernel case, the approximate kernel regression [1] with the prototype vectors formed by clustering all data with cardinality of 1000 was used on a large-scale SoF dataset for the proposed approach. On this dataset, the Nyström-based approximate kernel [36] was used for KSDA, KCDA, and KSMFA methods with cardinality 1000.

4.1. Single-view datasets

We conducted experiments on 4 facial image datasets, one large-scale facial image dataset, and 4 other datasets of various data types. The Jaffe [37] dataset contains facial images of Japanese females with 7 different facial expressions: anger, happiness, fear, disgust, sadness, surprise, and neutral. The dataset consists of 213 images and several examples can be seen from Fig. 3.

BU [38] dataset contains 700 images of individuals with the same 7 facial expressions.

The Cohn-Kanade [39] dataset contains 245 images of different people with different facial expressions of the same 7 classes as BU and Jaffe datasets. Example images can be seen in Fig. 4.

The Extended Yale-B dataset [40] contains 2432 grayscale facial images of 38 people and, therefore, defines a face recognition problem with 38 classes. Each class is represented by 64 images of

Fig. 3. An example of images from Jaffe dataset.

Fig. 4. Examples of images from Cohn-Kanade dataset.

Fig. 5. Examples of images from Yale dataset.
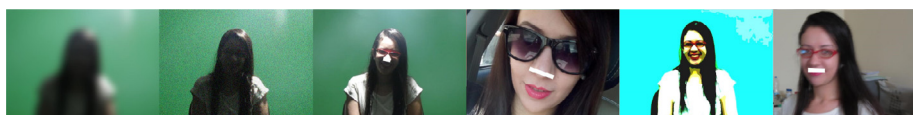

Fig. 6. Examples of images from SoF dataset.

the same person under different illumination conditions, positions, and view angles. Example images can be seen in Fig. 5.

The large-scale SoF dataset [41] consists of 42,592 images of 112 persons (66 male and 46 female) collected under different illumination conditions and containing images with occlusions (e.g. glasses). Example images can be seen in Fig. 6. All the facial image datasets mentioned above (i.e., Jaffe, Cohn-Kanade, BU, Yale-B, SoF) were reshaped to images of 30 × 40 pixels and flattened to obtain 1200 × 1 vectors.

The Ionosphere dataset [42] contains radar data represented as 351 34-dimensional vectors, along with the information on whether they contain evidence of some type of structure in the ionosphere or not, hence posing a binary classification problem. The Semeion dataset [43] contains 1593 instances of handwritten digits produced by 80 persons, each of whom had written each digit twice, in a normal way and in a fast way. The digits are represented by 16x16 binarized images flattened to 256 × 1 vectors.

The MONKS2 dataset [44] is derived from a domain, where each instance is represented by 6 discrete features corresponding to one of the two classes. The artificially generated data describes certain physical properties of robots, and the task is to predict the type of the robot based on these characteristics. The PIMA Indians Diabetes dataset [45] contains information on various medical attributes of patients, including the number of pregnancies the patient has had, their BMI, insulin level, age, along with the information on whether the patient has diabetes. The dataset contains 768 instances.

4.2. Multi-view datasets

For the evaluation of the multi-view methods seven datasets were used: Handwritten digits [46], Caltech-101 [47,48], NUS-WIDE [48,49], Human Action Recognition Using Smartphones [50], Robots Execution Failures [51], Healthy Old People Action Recognition [52], Million Song Dataset with Images (MSDI) [53]. The Handwritten digits dataset (HWD) [46] contains 2000 instances of handwritten digits of 10 classes. The images are represented by 6 views: Fourier coefficients (1 × 128), profile correlations (1 × 76), Karhunen-Love coefficients (1 × 64), pixel averages (1 × 240), Zernike moments (1 × 47), and morphological features (1 × 6).

The Caltech-101 dataset [47] is an image classification dataset represented by 6 views: Gabor features (1 × 48), wavelet moments (1 × 40), CENTRIST features (1 × 254), Histogram of oriented gradients features (1 × 1984), GIST features (1 × 512), local binary pattern features (1 × 928). Due to imbalanced data between classes, the dataset is divided into two subsets of 7 and 20 classes, resulting in 1474 and 2386 instances, respectively. Examples of images can be seen in Fig. 7.

NUS-WIDE dataset [49] is a large-scale image classification dataset of 31 classes described from 5 views: color histogram (1 × 65), color moments (1 × 226), color correlation (1 × 145), edge distribution (1 × 74), wavelet texture (1 × 129). Due to the large amount of samples, a subset of 11,288 instances is selected for the experiments. Examples of images can be seen in Fig. 8.

Fig. 7. Examples of images from Caltech-101 dataset.

Fig. 8. Examples of images from NUS-WIDE dataset.

Table 1

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

Dataset	SDA		CDA		SMFA		SRDA		SDA, sort. vec.		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	1	65.2	1	**66.2**	1
Ionosphere	**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 2

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

Dataset	SDA	CDA	SMFA	SRDA	SDA, sort. vec.	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**
Ionosphere	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	0.045	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**

The Human Action Recognition Using Smartphones dataset (HARS) [50] contains 3-axial angular velocity and linear acceleration data taken from the accelerometer and gyroscope data of a smart phone attached to a person's waist while the person is performing one of the 6 activities. Actions are described from 9 views: angular velocity of each of 3 axes, total acceleration of each of 3 axes, and body acceleration of each axis. Each view has the dimensionality of 128. Data was gathered from a group of 30 volunteers, resulting in 7352 instances. The cross-validation splits in our experiments were done such that the subjects performing the experiments are not repeated between training, validation, and test splits.

Healthy Old People Action Recognition dataset (HOPAR) [52] contains 2 datasets, each containing the information from a wireless sensor worn by a person, while performing one of the 4 activities: sitting on a bed, sitting on a chair, lying, ambulating. The data is organized into 4 views, where views 1–3 represent the acceleration from each of the 3 axes and view 4 contains information about the received signal strength indication, frequency, and phase of the signal, obtained from the sensor. The first dataset consists of data obtained from 60 subjects, out of which 25% of each class instances were selected, resulting in 10,495 instances. The second dataset contains information obtained from 27 subjects, resulting in 9057 instances.

The Robot Execution Failures dataset [51] consists of 5 subsets, each describing a different problem. For our experiments, subsets 1 and 4 were combined, resulting in a dataset of failures in approach to grasp or ungrasp position. The data is represented by 4 classes: normal, collision, frontal collision and obstruction, and described from 6 views: force on each of the 3 axes and torque on each of the 3 axes. Each view has 15 dimensions. The dataset consists of 205 instances.

The Million Song Dataset with Images (MSDI) [53] poses a music genre classification task for 15 different genres. Each instance represents a song, that is described from two views: audio spectrograms from audio signal and CNN features of the corresponding album cover. Both views have the dimensionality of 200. We perform evaluation on the subset of 7468 instances, chosen randomly from the dataset and preserving the initial class proportions.

4.3. Results

Tables 1 and 2 show the results for the single-view linear methods, where Table 1 depicts the accuracy and the number of subclasses resulting in the best accuracies, and Table 2 shows the training time in seconds. Tables 3 and 4 show similar information for the kernel methods. We performed experiments on 9 datasets using the proposed approach, which is compared to the conventional eigendecomposition-based approaches of SDA, CDA, SMFA

Table 3
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
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**

Table 5
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 6
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**

and with SRDA in the linear case; and KSDA, KCDA and KSMFA for the kernel case.

Tables 5 and 6 show the results for the multi-view case in linear formulations, while Tables 7 and 8 show the same information for kernel formulations. The results are presented similarly to Tables 1 and 2. The following methods are compared: single-view SDA, where features from different views are concatenated, MvMDA, and SMvDA. For the single-view SDA we use the proposed fast approach. We report the accuracy, time taken for training, and the number of subclasses that resulted in the highest accuracy. In the multi-view datasets, the clustering time is included in the total time, as the comparison is done with the methods that do not require clustering. In the single-view datasets, total time does not include the time used for clustering, as comparison is done

to other clustering-based methods, where the same subclass labels are used. It can be seen that the proposed single-view method is performing better or close to the conventional methods, while always taking less time.

Figs. 9 and 10 show the dependency of the training time on the number of training samples in the dataset based on training on the datasets used in this work: Fig. 9 depicts the single-view methods, and Fig. 10 depicts the multi-view methods. Note that the speed of the methods is dependent on multiple factors, including the dimensionality, the number of samples, number of classes, and subclasses. Besides, in multi-view cases, the number of views in the dataset affects the training time significantly. Therefore, the training time is not always increasing gradually with increase in dimensionality/number of samples, as can be observed especially in

Table 7
Classification results of kernel methods in multi-view datasets: accuracy/number of clusters per class.

Dataset	kernel SMvDA	kernel MvMDA	kernel MvSDA (our)		kernel 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 8
Classification results of kernel methods in multi-view datasets: training time (in sec).

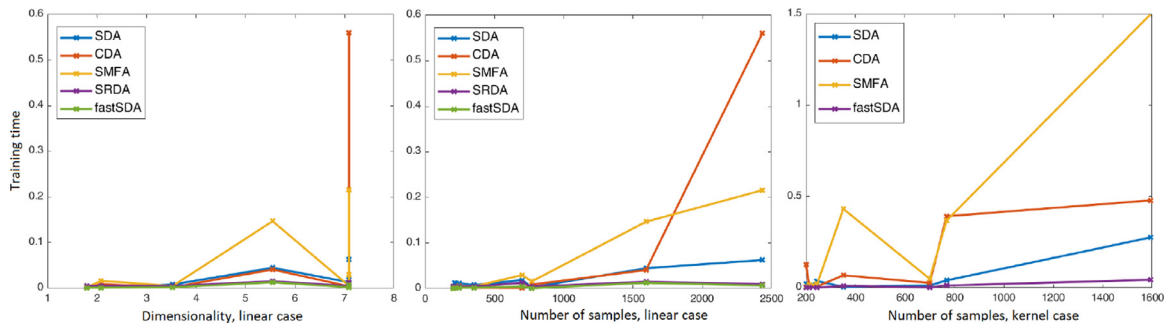| Dataset	kernel SMvDA	kernel MvMDA	kernel MvSDA (our)	kernel fastSDA (our)
HWD	72.4	70.5	5.7	**0.07**
HARS	561	554	97	**4.3**
Robots	0.14	0.14	0.03	**0.001**
Caltech-7	30.9	30	2.78	**0.03**
Caltech-20	244	236	9.57	**0.11**
HOPAR 1	76.7	74.4	10.9	**4.49**
HOPAR 2	65.9	74.1	8.89	**2.8**
MSDI	69.9	48.6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

Fig. 9. Dependency of training time on the dimensionality of data (left, dimensionality shown on log scale) and dependency of training time on the number of samples in the linear methods (middle) and kernel methods (right).
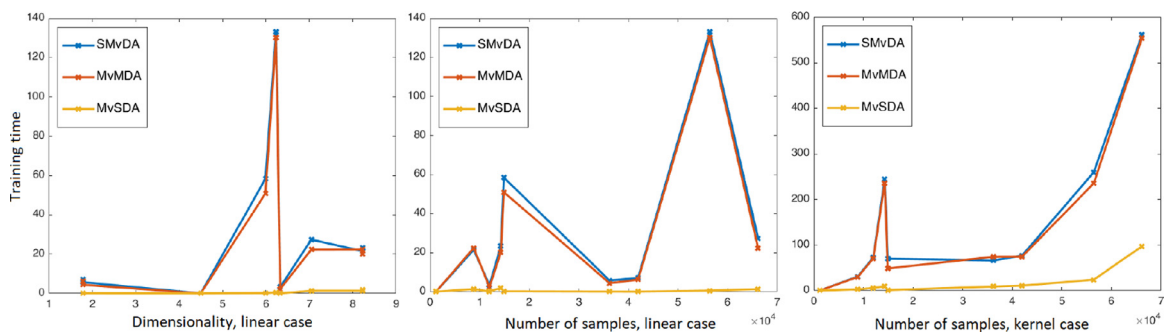

Fig. 10. Dependency of training time on the dimensionality of data (left, dimensionality shown on log scale) and on the number of samples (middle) in the linear multi-view methods and dependency of training time on the number of samples in the kernel multi-view methods (right). Numbers of samples and dimensionalities are summed across the views.

the plots corresponding to linear formulations. However, it can be observed that the proposed methods outperform the existing approaches in terms of training time and the margin becomes higher with larger dataset sizes and dimensionalities. This can be seen especially well from the kernel formulation plots.

In the single-view linear case, the training time of fastSDA is similar to that of SRDA. However, accuracy-wise the proposed approach outperforms SRDA because of SRDA's assumptions on uni-

modality. This can be seen from Table 1. Otherwise, in the kernel formulation and in the multi-view scenarios, the proposed approach outperforms other methods in terms of computational complexity by a significant margin.

In addition, by performing the projection onto the sorted by criterion value (11) regressed eigenvectors of L_b, we verify that for the data with subclass structure the eigenvectors corresponding to larger criterion values are those following the described struc-

ture. The only exceptions were observed in the Monks2 and PIMA datasets, where some of the eigenvectors had random structure - this is due to the samples of different subclasses being mixed with each other. However, even in this case, it can be observed that the proposed approach results in competitive accuracy and higher speed. The accuracy obtained by projecting data using the transformation matrix comprised of eigenvectors corresponding to largest criterion values is shown in the second last column of Table 1.

For the multi-view case we compared the proposed multi-view SDA to other multi-view methods that assume unimodality of data. It can be seen that the proposed approach results in significant speed-up and competitive accuracy, often outperforming competing methods.

5. Conclusions

This work presents two contributions, proposing a fast and efficient solution for Subclass Discriminant Analysis and introducing multi-view Subclass Discriminant Analysis with a fast solution. As can be seen from the experimental results, the proposed speed-up approach allows to reduce the training time significantly, while being competitive in accuracy and often outperforming the conventional methods. Our findings allow performing the analysis on large-scale datasets, where conventional solutions are not feasible. The proposed multi-view Subclass Discriminant Analysis provides superior accuracy compared to the methods relying on the assumption of unimodality of data. In addition, the proposed speed-up approach can be applied to this formulation, resulting in a significant gain in speed.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

References

[1] A. Iosifidis, M. Gabbouj, Scaling up class-specific kernel discriminant analysis for large-scale face verification, IEEE Trans. Inf. Forensics Secur. 11 (2016) 2453–2465.

[2] C. Zhao, X. Wang, D. Miao, H. Wang, W. Zheng, Y. Xu, D. Zhang, Maximal granularity structure and generalized multi-view discriminant analysis for person re-identification, Pattern Recognit. 79 (2018) 79–96.

[3] R. Duda, P. Hart, D. Stork, Pattern Classification, second ed., Wiley, New York, NY, USA, 2000.

[4] J. Ye, Least squares linear discriminant analysis, Int. Conf. Mach. Learn. 1 (2007) 1087–1093.

[5] H. Wang, X. Lu, W. Zheng, Fisher discriminant analysis with l1-norm, IEEE Trans. Cybern. 4 (2014) 828–842.

[6] R. Fisher, The statistical utilization of multiple measurements, Ann. Eugen. 8 (1938) 376–386.

[7] M. Zhu, A. Martinez, Subclass discriminant analysis, IEEE Trans. Pattern Anal. Mach. Intell. 28 (2006) 1274–1286.

[8] X. Chen, T. Huang, Facial expression recognition: a clustering-based approach, Pattern Recognit. Lett. 24 (2003) 1295–1302.

[9] A. Maronidis, A. Tefas, I. Pitas, Subclass graph embedding and a marginal fisher analysis paradigm, Pattern Recognit. 48 (2015) 4024–4035.

[10] N. Kwak, Implementing kernel methods incrementally by incremental nonlinear projection trick, IEEE Trans. Cybern. 47 (2017) 4003–4009.

[11] D. Cai, X. He, J. Han, Speed up kernel discriminant analysis, VLDB J. 20 (2011) 21–33.

[12] A. Iosifidis, M. Gabbouj, Class-specific kernel discriminant analysis revisited: further analysis and extensions, IEEE Trans. Cybern. 47 (2017) 4485–4496.

[13] A. Iosifidis, M. Gabbouj, On the kernel extreme learning machine speedup, Pattern Recognit. Lett. 68 (2015) 205–210.

[14] H. Ye, Y. Li, C. Chen, Z. Zhang, Fast fisher discriminant analysis with randomized algorithms, Pattern Recognit. 72 (2017) 82–92.

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

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

[17] A. Maronidis, A. Tefas, I. Pitas, Graph embedding exploiting subclasses, in: IEEE Symposium Series on Computational Intelligence, Cape Town, South Africa, 2015, pp. 1452–1459.

[18] M. Kan, S. Shan, H. Zhang, S. Lao, X. Chen, Multi-view discriminant analysis, IEEE Trans. Pattern Anal. Mach. Intell. 38 (2016) 188194.

[19] X. You, J. Xu, W. Yuan, X. Jing, D. Tao, T. Zhang, Multi-view common component discriminant analysis for cross-view classification, Pattern Recognit. 92 (2019) 37–51.

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

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

[22] G. Cao, A. Iosifidis, K. Chen, M. Gabbouj, Generalized multi-view embedding for visual recognition and cross-modal retrieval, IEEE Trans. Cybern. 48 (2017) 2542–2555.

[23] A. Iosifidis, A. Tefas, I. Pitas, On the optimal class representation in linear discriminant analysis, IEEE Trans. Neural Netw. Learn. Syst. 24 (9) (2013) 1491–1497.

[24] A. Iosifidis, A. Tefas, I. Pitas, Kernel reference discriminant analysis, Pattern Recognit. Lett. 49 (2014) 85–91.

[25] A. Iosifidis, A. Tefas, I. Pitas, Class-specific reference discriminant analysis with application in human behavior analysis, IEEE Trans. Hum. Mach. Syst. 45 (2015) 315–326.

[26] Y. Shuicheng, X. Dong, B. Zhang, H. Zhang, Q. Yang, S. Lin, Graph embedding and extensions: a general framework for dimensionality reduction, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2007) 40–51.

[27] B. Scholkopf., S. Mika, C. Burges, P. Knirsch, K. Muller, G. Ratsch, A. Smola, Input space versus feature space in kernel-based methods, IEEE Trans. Neural Netw. (1999) 10001017.

[28] B. Scholkopf, A. Smola, K. Muller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Comput. 10 (1998) 12991319.

[29] D. Cai, X. He, J. Han, Srda: an efficient algorithm for large scale discriminant analysis, IEEE Trans. Knowl. Data Eng. 20 (2007) 1–12.

[30] M. Idaji, M. Shamsollahi, S. Sardouie, Higher order spectral regression discriminant analysis (HOSRDA): a tensor feature reduction method for ERP detection, Pattern Recognit. 70 (2017) 152–162.

[31] C. Paige, M. Saunders, LSQR: an algorithm for sparse linear equations and sparse least squares, ACM Trans. Math. Softw. 8 (1982) 43–71.

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

[33] D. Cai, X. He, J. Han, Spectral regression for efficient regularized subspace learning, in: IEEE International Conference on Computer Vision, Rio de Janeiro, Brazil, 2007, pp. 1–8.

[34] R. Horn, C. Johnson, Matrix Analysis, first ed., Cambridge University Press, 1985.

[35] A. Ruhe, Matrix algorithms volume 1: basic decompositions, 2000.

[36] A. Iosifidis, M. Gabbouj, Nyström-based approximate subspace learning, Pattern Recognit. 57 (2016) 190–197.

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

[38] L. Yin, X. Wei, Y. Sun, J. Wang, M. Rosato, A 3d facial expression database for facial behavior research, in: IEEE International Conference on Automatic Face and Gesture Recognition, Southampton, UK, 2006, pp. 211–216.

[39] T. Kanade, J. Cohn, Y. Tian, Comprehensive database for facial expression analysis, in: IEEE International Conference on Automatic Face and Gesture Recognition, Grenoble, France, 2000, pp. 46–53.

[40] K. Lee, J. Ho, D. Kriegman, Acquiring linear subspaces for face recognition under variable lightning, IEEE Trans. Pattern Anal. Mach. Intell. 27 (2005) 684–698.

[41] M. Afifi, 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).

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

[43] M. Buscema, Metanet: the theory of independent judges, Subst. Use Misuse 33 (1998) 439–461.

[44] J. Wnek, R. Michalski, Comparing symbolic and subsymbolic learning: three studies, Mach. Learn. A Multistrategy Approach 4 (1993).

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

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

[47] L. Fei-Fei, R. Fergus, P. Perona, One-shot learning of object categories, IEEE Trans. Pattern Recognit.Mach. Intell. 8 (2006) 594–611.

[48] Y. Li, F. Nie, H. Huang, J. Huang, Large-scale multi-view spectral clustering via bipartite graph, in: AAAI Conference on Artificial Intelligence, 2015, pp. 2750–2756.

[49] T. Chua, J. Tang, R. Hong, H. Li, Z. Luo, Y. Zheng, NUS-WIDE: a real-world web image database from National University of Singapore, in: ACM International Conference on Image and Video Retrieval, Greece, 2009, p. 48.

[50] D. Anguita, A. Ghio, L. Oneto, X. Parra, J. Reyes-Ortiz, A public domain dataset for human activity recognition using smartphones, in: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, Bruges, Belgium, 2013, pp. 437–442.

[51] M. Afifi, A. Abdelhamed, Integration and learning in supervision of flexible assembly systems, IEEE Trans. Rob. Autom. 12 (1996) 202–219.

[52] S. Torres, D. Ranasinghe, Q. Shi, A. Sample, Sensor enabled wearable RFID technology for mitigating the risk of falls near beds, in: IEEE International Conference on RFID, 2013, pp. 191–198.

[53] S. Oramas, F. Barbieri, O. Nieto, Multimodal deep learning for music genre classification, Trans. Int. Soc. Music Inf. Retr. 1 (2018) 4–21.

Kateryna Chumachenko received her B.Eng. degree from South-Eastern Finland University of Applied Sciences, Mikkeli, Finland, and Kharkiv National University of Radio Electronics, Ukraine, in 2017 and 2018, respectively. She is currently obtaining a Master of Science degree within department of Computing Sciences, Tampere University, Finland.

Jenni Raitoharju is a postdoctoral research fellow in Unit of Computing Sciences, Tampere University, Finland. She received her Ph.D. in Information Technology in Tampere University of Technology in 2017. Her current projects deal with machine learning and pattern recognition in applications such as biomonitoring, intelligent buildings, and autonomous boats.

Alexandros Iosifidis received his Ph.D. degree in Informatics from the Aristotle University of Thessaloniki in 2014. He an Associate Professor of Machine Learning at Aarhus University, Denmark. His research interests include statistical machine learning and artificial neural networks with applications in Computer Vision and time-series analysis problems.

Moncef Gabbouj received his MS and Ph.D. degrees in electrical engineering from Purdue University, in 1986 and 1989, respectively. Dr. Gabbouj is Professor of Signal Processing at the Department of Computing Sciences, Tampere University, Finland. His research interests include Big Data analytics, multimedia analysis, artificial intelligence, machine learning, pattern recognition.