# Linear Discriminant Analysis: A Detailed Tutorial

Alaa Tharwat *
*Department of Computer Science and Engineering, Frankfurt University of Applied Sciences, Frankfurt am Main, Germany*
*Faculty of Engineering, Suez Canal University, Egypt*
*E-mail: engalaatharwat@hotmail.com*

Tarek Gaber *
*Faculty of Computers and Informatics, Suez Canal University, Egypt*
*E-mail: tmgaber@gmail.com*

Abdelhameed Ibrahim *
*Faculty of Engineering, Mansoura University, Egypt*
*Email: afai79@yahoo.com*

Aboul Ella Hassanien*
*Faculty of Computers and Information, Cairo University, Egypt*
*E-mail: aboitcairo@gmail.com*
**Scientific Research Group in Egypt, (SRGE), http://www.egyptscience.net*

Linear Discriminant Analysis (LDA) is a very common technique for dimensionality reduction problems as a pre-processing step for machine learning and pattern classification applications. At the same time, it is usually used as a black box, but (sometimes) not well understood. The aim of this paper is to build a solid intuition for what is LDA, and how LDA works, thus enabling readers of all levels be able to get a better understanding of the LDA and to know how to apply this technique in different applications. The paper first gave the basic definitions and steps of how LDA technique works supported with visual explanations of these steps. Moreover, the two methods of computing the LDA space, i.e. *class-dependent* and *class-independent* methods, were explained in details. Then, in a step-by-step approach, two numerical examples are demonstrated to show how the LDA space can be calculated in case of the class-dependent and class-independent methods. Furthermore, two of the most common LDA problems (i.e. *Small Sample Size* (*SSS*) and non-linearity problems) were highlighted and illustrated, and state-of-the-art solutions to these problems were investigated and explained. Finally, a number of experiments was conducted with different datasets to (1) investigate the effect of the eigenvectors that used in the LDA space on the robustness of the extracted feature for the classification accuracy, and (2) to show when the *SSS* problem occurs and how it can be addressed.

Keywords: Dimensionality reduction, PCA, LDA, Kernel Functions, Class-Dependent LDA, Class-Independent LDA, SSS (Small Sample Size) problem,, eigenvectors artificial intelligence

## 1. Introduction

Dimensionality reduction techniques are important in many applications related to machine learning [15], data mining [6,33], Bioinformatics [47], biometric [61] and information retrieval [73]. The main goal of the dimensionality reduction techniques is to reduce the dimensions by removing the redundant and dependent features by transforming the features from a higher dimensional space that may lead to a curse of dimensionality problem, to a space with lower dimensions. There are two major approaches of the dimensionality reduction techniques, namely, unsupervised and supervised approaches. In the unsupervised approach, there is no need for labeling classes of the data. While in the supervised approach, the dimensionality reduction techniques take the class labels into consideration [32,15].

There are many unsupervised dimensionality reduction techniques such as Independent Component Analysis (ICA) [31,28] and Non-negative Matrix Factorization (NMF) [14], but the most famous technique of the unsupervised approach is the Principal Component Analysis (PCA) [71,4,67,62]. This type of data reduction is suitable for many applications such as visualization [40,2], and noise removal [70]. On the other hand, the supervised approach has many techniques such as Mixture Discriminant Analysis (MDA) [25] and Neural Networks (NN) [27], but the most famous technique of this approach is the Linear Discriminant Analysis (LDA) [50]. This category of dimensionality reduction

techniques are used in biometrics [36,12], Bioinformatics [77], and chemistry [11].

The LDA technique is developed to transform the features into a lower dimensional space, which maximizes the ratio of the between-class variance to the within-class variance, thereby guaranteeing maximum class separability [76,43]. There are two types of LDA technique to deal with classes: *class-dependent* and *class-independent*. In the class-dependent LDA, one separate lower dimensional space is calculated for each class to project its data on it whereas, in the class-independent LDA, each class will be considered as a separate class against the other classes [1,74]. In this type, there is just one lower dimensional space for all classes to project their data on it.

Although the LDA technique is considered the most well-used data reduction techniques, it suffers from a number of problems. In the first problem, LDA fails to find the lower dimensional space if the dimensions are much higher than the number of samples in the data matrix. Thus, the within-class matrix becomes singular, which is known as the *small sample problem* (SSS). There are different approaches that proposed to solve this problem. The first approach is to remove the null space of within-class matrix as reported in [79,56]. The second approach used an intermediate subspace (e.g. PCA) to convert a within-class matrix to a full-rank matrix; thus, it can be inverted [35,4]. The third approach, a well-known solution, is to use the regularization method to solve a singular linear systems [38,57]. In the second problem, the linearity problem, if different classes are non-linearly separable, the LDA cannot discriminate between these classes. One solution to this problem is to use the kernel functions as reported in [50].

The brief tutorials on the two LDA types are reported in [1]. However, the authors did not show the LDA algorithm in details using numerical tutorials, visualized examples, nor giving insight investigation of experimental results. Moreover, in [57], an overview of the SSS for the LDA technique was presented including the theoretical background of the SSS problem. Moreover, different variants of LDA technique were used to solve the SSS problem such as Direct LDA (DLDA) [22,83], regularized LDA (RLDA) [18,37,38,80], PCA+LDA [42], Null LDA (NLDA) [10,82], and kernel DLDA (KDLDA) [36]. In addition, the authors presented different applications that used the LDA-SSS techniques such as face recognition and cancer classification. Furthermore, they conducted different experiments using three well-known face recognition datasets to compare between different variants of the LDA technique. Nonetheless, in [57], there is no detailed explanation of how (with numerical examples) to calculate the within and between class variances to construct the LDA space. In addition, the steps of constructing the LDA space are not supported with well-explained graphs helping for well understanding of the LDA underlying mechanism. In addition, the non-linearity problem was not highlighted.

This paper gives a detailed tutorial about the LDA technique, and it is divided into five sections. Section 2 gives an overview about the definition of the main idea of the LDA and its background. This section begins by explaining how to calculate, with visual explanations, the between-class variance, within-class variance, and how to construct the LDA space. The algorithms of calculating the LDA space and projecting the data onto this space to reduce its dimension are then introduced. Section 3 illustrates numerical examples to show how to calculate the LDA space and how to select the most robust eigenvectors to build the LDA space. While Section 4 explains the most two common problems of the LDA technique and a number of state-of-the-art methods to solve (or approximately solve) these problems. Different applications that used LDA technique are introduced in Section 5. In Section 6, different packages for the LDA and its variants were presented. In Section 7, two experiments are conducted to show (1) the influence of the number of the selected eigenvectors on the robustness and dimension of the LDA space, (2) how the *SSS* problem occurs and highlights the well-known methods to solve this problem. Finally, concluding remarks will be given in Section 8.

## 2. LDA Technique

### *2.1. Definition of LDA*

The goal of the LDA technique is to project the original data matrix onto a lower dimensional space. To achieve this goal, three steps needed to be performed. The first step is to calculate the separability between different classes (i.e. the distance between the means of different classes), which is called the *between-class variance* or *between-class matrix*. The second step is to calculate the distance between the mean and the samples of each class, which is called the *within-class variance* or *within-class matrix*. The third step is to construct the lower dimensional space which maximizes the between-class variance and minimizes the within-

class variance. This section will explain these three steps in detail, and then the full description of the LDA algorithm will be given. Figures (1 and 2) are used to visualize the steps of the LDA technique.

### 2.2. Calculating the Between-Class Variance ($S_B$)

The between-class variance of the $i^{th}$ class ($S_{B_i}$) represents the distance between the mean of the $i^{th}$ class ($\mu_i$) and the total mean ($\mu$). LDA technique searches for a lower-dimensional space, which is used to maximize the between-class variance, or simply maximize the separation distance between classes. To explain how the between-class variance or the between-class matrix ($S_B$) can be calculated, the following assumptions are made. Given the original data matrix $X = \{x_1, x_2, \ldots, x_N\}$, where $x_i$ represents the $i^{th}$ sample, pattern, or observation and $N$ is the total number of samples. Each sample is represented by $M$ features ($x_i \in \mathcal{R}^M$). In other words, each sample is represented as a point in $M$-dimensional space. Assume the data matrix is partitioned into $c = 3$ classes as follows, $X = [\omega_1, \omega_2, \omega_3]$ as shown in Fig. (1, step (**A**)). Each class has five samples (i.e. $n_1 = n_2 = n_3 = 5$), where $n_i$ represents the number of samples of the $i^{th}$ class. The total number of samples ($N$) is calculated as follows, $N = \sum_{i=1}^{3} n_i$.

To calculate the between-class variance ($S_B$), the separation distance between different classes which is denoted by $(m_i - m)$ will be calculated as follows:

$$(m_i - m)^2 = (W^T \mu_i - W^T \mu)^2 = W^T (\mu_i - \mu)(\mu_i - \mu)^T W \tag{1}$$

where $m_i$ represents the projection of the mean of the $i^{th}$ class and it is calculated as follows, $m_i = W^T \mu_i$, where $m$ is the projection of the total mean of all classes and it is calculated as follows, $m = W^T \mu$, $W$ represents the transformation matrix of LDA[1], $\mu_i (1 \times M)$ represents the mean of the $i^{th}$ class and it is computed as in Equation (2), and $\mu (1 \times M)$ is the total mean of all classes and it can be computed as in Equation (3) [83,36]. Figure (1) shows the mean of each class and the total mean in step (**B** and **C**), respectively.

$$\mu_j = \frac{1}{n_j} \sum_{x_i \in \omega_j} x_i \tag{2}$$

[1]The transformation matrix ($W$) will be explained in Sect. 2.4

$$\mu = \frac{1}{N} \sum_{i=1}^{N} x_i = \sum_{i=1}^{c} \frac{n_i}{N} \mu_i \tag{3}$$

where $c$ represents the total number of classes (in our example $c = 3$).

The term $(\mu_i - \mu)(\mu_i - \mu)^T$ in Equation (1) represents the separation distance between the mean of the $i^{th}$ class ($\mu_i$) and the total mean ($\mu$), or simply it represents the between-class variance of the $i^{th}$ class ($S_{B_i}$). Substitute $S_{B_i}$ into Equation (1) as follows:

$$(m_i - m)^2 = W^T S_{Bi} W \tag{4}$$

The total between-class variance is calculated as follows, ($S_B = \sum_{i=1}^{c} n_i S_{B_i}$). Figure (1, step (**D**)) shows first how the between-class matrix of the first class ($S_{B_1}$) is calculated and then how the total between-class matrix ($S_B$) is then calculated by adding all the between-class matrices of all classes.

### 2.3. Calculating the Within-Class Variance ($S_W$)

The within-class variance of the $i^{th}$ class ($S_{W_i}$) represents the difference between the mean and the samples of that class. LDA technique searches for a lower-dimensional space, which is used to minimize the difference between the projected mean ($m_i$) and the projected samples of each class ($W^T x_i$), or simply minimizes the within-class variance [83,36]. The within-class variance of each class ($S_{W_j}$) is calculated as in Equation (5).

$$\begin{aligned} \sum_{x_i \in \omega_j, j=1,\ldots,c} & (W^T x_i - m_j)^2 \\ &= \sum_{x_i \in \omega_j, j=1,\ldots,c} (W^T x_{ij} - W^T \mu_j)^2 \\ &= \sum_{x_i \in \omega_j, j=1,\ldots,c} W^T (x_{ij} - \mu_j)^2 W \\ &= \sum_{x_i \in \omega_j, j=1,\ldots,c} W^T (x_{ij} - \mu_j)(x_{ij} - \mu_j)^T W \\ &= \sum_{x_i \in \omega_j, j=1,\ldots,c} W^T S_{W_j} W \end{aligned} \tag{5}$$

From Equation (5), the within-class variance for each class can be calculated as follows, $S_{W_j} = d_j^T * d_j = \sum_{i=1}^{n_j} (x_{ij} - \mu_j)(x_{ij} - \mu_j)^T$, where $x_{ij}$ represents the $i^{th}$ sample in the $j^{th}$ class as shown in Fig. (1, step (**E,**

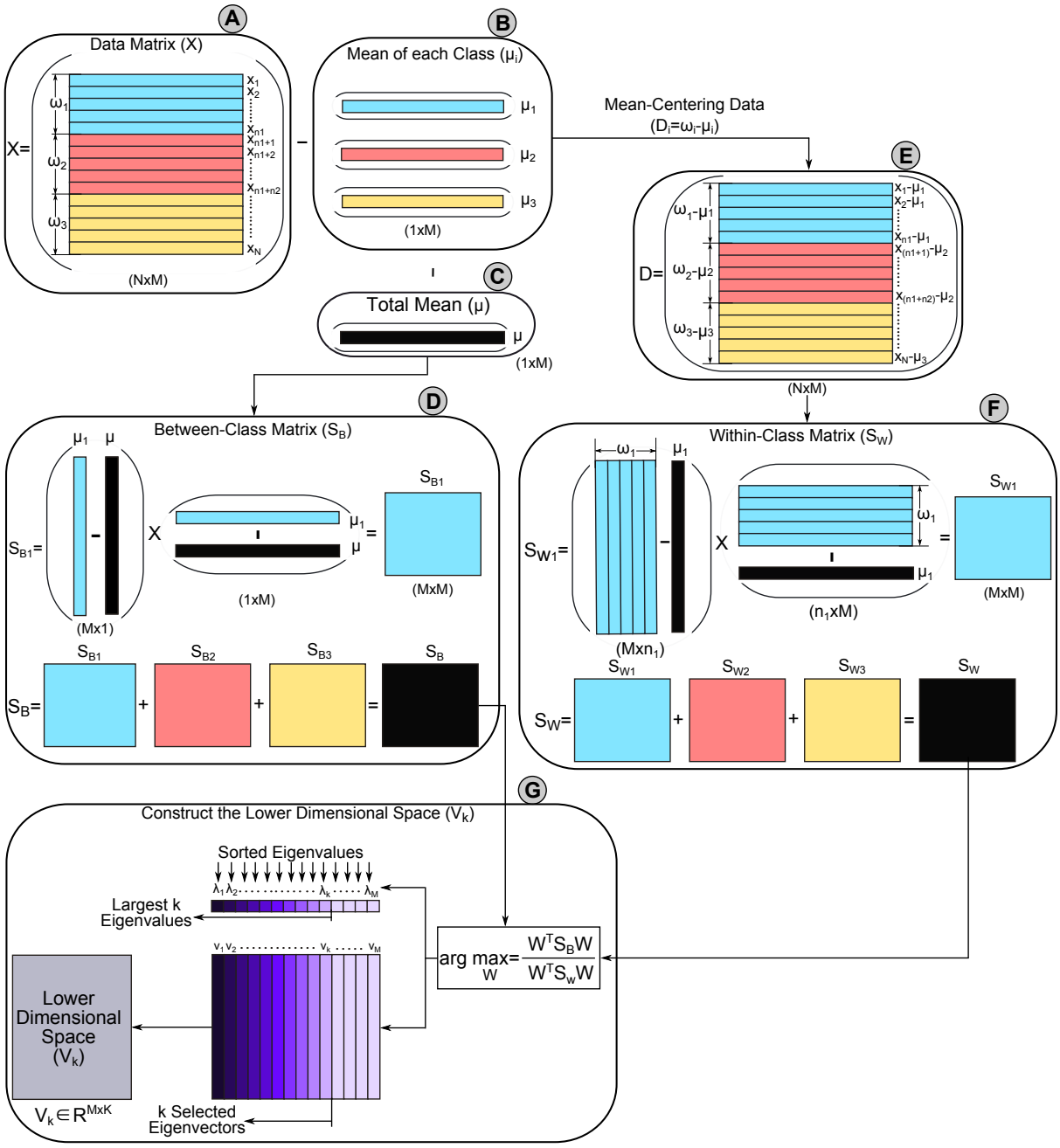



Fig. 1. Visualized steps to calculate a lower dimensional subspace of the LDA technique.

**F**)), and $d_j$ is the centering data of the $j^{th}$ class, i.e. $d_j = \omega_j - \mu_j = \{x_i\}_{i=1}^{n_j} - \mu_j$. Moreover, step (**F**) in the figure illustrates how the within-class variance of the first class ($S_{W_1}$) in our example is calculated. The total within-class variance represents the sum of all within-class matrices of all classes (see Fig. (1, step (**F**))), and it can be calculated as in Equation (6).

Table 1
Notation.

| Notation | Description | Notation | Description |
|---|---|---|---|
| $X$ | Data matrix | $x_i$ | $i^{th}$ sample |
| $N$ | Total number of samples in $X$ | $M$ | Dimension of $X$ or the number of features of $X$ |
| $W$ | Transformation matrix | $V_k$ | The lower dimensional space |
| $n_i$ | Number of samples in $\omega_i$ | $c$ | Total number of classes |
| $\mu_i$ | The mean of the $i^{th}$ class | $m_i$ | The mean of the $i^{th}$ class after projection |
| $\mu$ | Total or global mean of all samples | $m$ | The total mean of all classes after projection |
| $S_{Wi}$ | Within-class variance or scatter matrix of the $i^{th}$ class ($\omega_i$) | $S_W$ | Within-class variance |
| $S_{Bi}$ | Between-class variance of the $i^{th}$ class ($\omega_i$) | $S_B$ | Between-class variance |
| $V$ | Eigenvectors of $W$ | $\lambda$ | Eigenvalue matrix |
| $V_i$ | $i^{th}$ eigenvector | $\lambda_i$ | $i^{th}$ eigenvalue |
| $x_{ij}$ | The $i^{th}$ sample in the $j^{th}$ class | $Y$ | Projection of the original data |
| $k$ | The dimension of the lower dimensional space ($V_k$) | $\omega_i$ | $i^{th}$ Class |

$$
\begin{aligned}
S_W &= \sum_{i=1}^{3} S_{W_i} \\
&= \sum_{x_i \in \omega_1} (x_i - \mu_1)(x_i - \mu_1)^T \\
&+ \sum_{x_i \in \omega_2} (x_i - \mu_2)(x_i - \mu_2)^T \\
&+ \sum_{x_i \in \omega_3} (x_i - \mu_3)(x_i - \mu_3)^T
\end{aligned}
\tag{6}
$$

### 2.4. Constructing the Lower Dimensional Space

After calculating the between-class variance ($S_B$) and within-class variance ($S_W$), the transformation matrix ($W$) of the LDA technique can be calculated as in Equation (7), which is called Fisher's criterion. This formula can be reformulated as in Equation (8).

$$
arg\ \max_{W} \frac{W^T S_B\, W}{W^T S_W\, W} \tag{7}
$$

$$
S_W\, W = \lambda S_B\, W \tag{8}
$$

where $\lambda$ represents the eigenvalues of the transformation matrix ($W$). The solution of this problem can be obtained by calculating the eigenvalues ($\lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_M\}$) and eigenvectors ($V = \{v_1, v_2, \ldots, v_M\}$) of $W = S_W^{-1} S_B$, if $S_W$ is non-singular [83,36,81].

The eigenvalues are scalar values, while the eigenvectors are non-zero vectors, which satisfies the Equation (8) and provides us with the information about the LDA space. The eigenvectors represent the directions of the new space, and the corresponding eigenvalues represent the scaling factor, length, or the magnitude of the eigenvectors [59,34]. Thus, each eigenvector represents one axis of the LDA space, and the associated eigenvalue represents the robustness of this eigenvector. The robustness of the eigenvector reflects its ability to discriminate between different classes, i.e. increase the between-class variance, and decreases the within-class variance of each class; hence meets the LDA goal. Thus, the eigenvectors with the $k$ highest eigenvalues are used to construct a lower dimensional space ($V_k$), while the other eigenvectors ($\{v_{k+1}, v_{k+2}, v_M\}$) are neglected as shown in Fig. (1, step (**G**)).

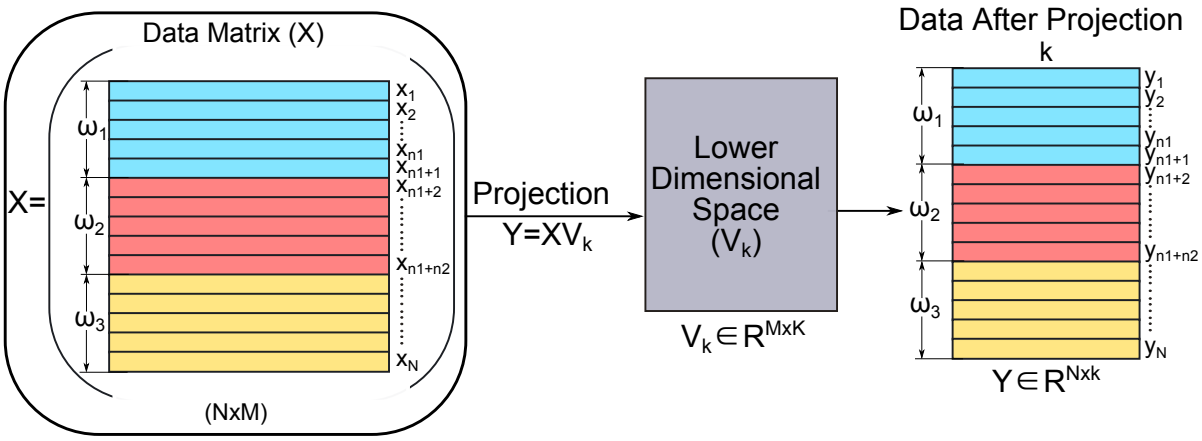



Fig. 2. Projection of the original samples (i.e. data matrix) on the lower dimensional space of LDA ($V_k$).

Figure (2) shows the lower dimensional space of the LDA technique, which is calculated as in Fig. (1, step (**G**)). As shown, the dimension of the original data matrix ($X \in \mathcal{R}^{N\times M}$) is reduced by projecting it onto the lower dimensional space of LDA ($V_k \in \mathcal{R}^{M\times k}$) as denoted in Equation (9) [81]. The dimension of the data after projection is $k$; hence, $M-k$ features are ignored or deleted from each sample. Thus, each sample ($x_i$) which was represented as a point a $M$-dimensional space will be represented in a $k$-dimensional space by projecting it onto the lower dimensional space ($V_k$) as follows, $y_i = x_i V_k$.

$$Y = XV_k \tag{9}$$

Figure (3) shows a comparison between two lower-dimensional sub-spaces. In this figure, the original data which consists of three classes as in our example are plotted. Each class has five samples, and all samples are represented by two features only ($x_i \in \mathcal{R}^2$) to be visualized. Thus, each sample is represented as a point in two-dimensional space. The transformation matrix ($W(2\times 2)$) is calculated using the steps in Sect. 2.2, 2.3, and 2.4. The eigenvalues ($\lambda_1$ and $\lambda_2$) and eigenvectors (i.e. sub-spaces) ($V = \{v_1, v_2\}$) of $W$ are then calculated. Thus, there are two eigenvectors or sub-spaces. A comparison between the two lower-dimensional sub-spaces shows the following notices:

– First, the separation distance between different classes when the data are projected on the first eigenvector ($v_1$) is much greater than when the data are projected on the second eigenvector ($v_2$). As shown in the figure, the three classes are efficiently discriminated when the data are projected on $v_1$. Moreover, the distance between the means of the first and second classes ($m_1 - m_2$) when the original data are projected on $v_1$ is much greater than when the data are projected on $v_2$, which reflects that the first eigenvector discriminates the three classes better than the second one.

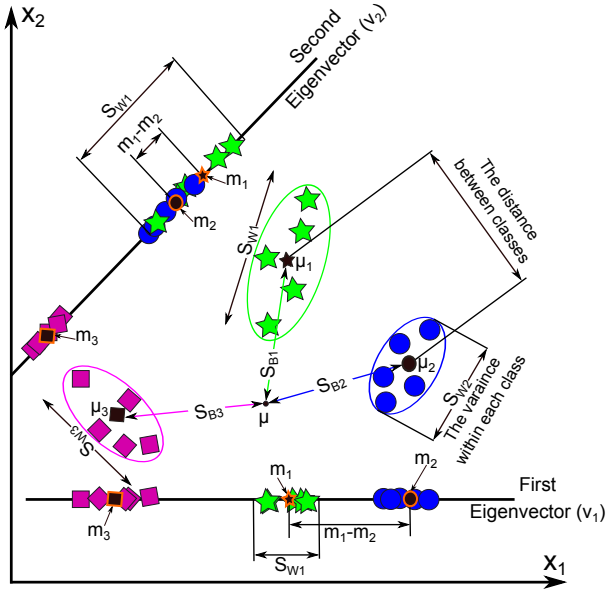



Fig. 3. A visualized comparison between the two lower-dimensional sub-spaces which are calculated using three different classes.

– Second, the within-class variance when the data are projected on $v_1$ is much smaller than when it projected on $v_2$. For example, $S_{W_1}$ when the data are projected on $v_1$ is much smaller than when the data are projected on $v_2$. Thus, projecting the data on $v_1$ minimizes the within-class variance much better than $v_2$.

From these two notes, we conclude that the first eigenvector meets the goal of the lower-dimensional space of the LDA technique than the second eigenvector; hence, it is selected to construct a lower-dimensional space.

### 2.5. Class-Dependent vs. Class-Independent Methods

The aim of the two methods of the LDA is to calculate the LDA space. In the class-dependent LDA, one separate lower dimensional space is calculated for each class as follows, $W_i = S_{W_i}^{-1} S_B$, where $W_i$ represents the transformation matrix for the i$^{th}$ class. Thus, eigenvalues and eigenvectors are calculated for each transformation matrix separately. Hence, the samples of each class are projected on their corresponding eigenvectors. On the other hand, in the class-independent method, one lower dimensional space is calculated for all classes. Thus, the transformation matrix is calculated for all classes, and the samples of all classes are projected on the selected eigenvectors [1].

### 2.6. LDA Algorithm

In this section the detailed steps of the algorithms of the two LDA methods are presented. As shown in Algorithms (1 and 2), the first four steps in both algorithms are the same. Table (1) shows the notations which are used in the two algorithms.

### 2.7. Computational Complexity of LDA

In this section, the computational complexity for LDA is analyzed. The computational complexity for the first four steps, common in both class-dependent and class-independent methods, are computed as follows. As illustrated in Algorithm (1), in step (2), to calculate the mean of the $i^{th}$ class, there are $n_i M$ additions and $M$ divisions, i.e., in total, there are $(NM + cM)$ operations. In step (3), there are $NM$ additions and $M$ divisions, i.e., there are $(NM + M)$ operations. The computational complexity of the fourth step is $c(M + M^2 + M^2)$, where $M$ is for $\mu_i - \mu$, $M^2$ for $(\mu_i - \mu)(\mu_i - \mu)^T$, and the last $M^2$ is for the multiplication between $n_i$ and the matrix $(\mu_i - \mu)(\mu_i - \mu)^T$. In the fifth step, there are $N(M + M^2)$ operations, where $M$ is for $(x_{ij} - \mu_j)$ and $M^2$ is for $(x_{ij} - \mu_j)(x_{ij} - \mu_j)^T$. In the sixth step, there are $M^3$ operations to calculate $S_W^{-1}$, $M^3$ is for the multiplication between $S_W^{-1}$ and $S_B$, and $M^3$ to calculate the eigenvalues and eigenvectors. Thus, in class-independent method, the computational complexity is $O(NM^2)$ if $N > M$; otherwise, the complexity is $O(M^3)$.

In Algorithm (2), the number of operations to calculate the within-class variance for each class $S_{W_j}$ in the sixth step is $n_j(M + M^2)$, and to calculate $S_W$, $N(M + M^2)$ operations are needed. Hence, calculating the within-class variance for both LDA methods are the same. In the seventh step and eighth, there are $M^3$ operations for the inverse, $M^3$ for the multiplication of $S_{W_i}^{-1} S_B$, and $M^3$ for calculating eigenvalues and eigenvectors. These two steps are repeated for

**Algorithm 1.** : Linear Discriminant Analysis (LDA): Class-Independent

1: Given a set of $N$ samples $[x_i]_{i=1}^N$, each of which is represented as a row of length $M$ as in Fig. (1, step(**A**)), and $X(N \times M)$ is given by,

$$X = \begin{bmatrix} x_{(1,1)} & x_{(1,2)} & \cdots & x_{(1,M)} \\ x_{(2,1)} & x_{(2,2)} & \cdots & x_{(2,M)} \\ \vdots & \vdots & \vdots & \vdots \\ x_{(N,1)} & x_{(N,2)} & \cdots & x_{(N,M)} \end{bmatrix} \tag{10}$$

2: Compute the mean of each class $\mu_i(1 \times M)$ as in Equation (2).

3: Compute the total mean of all data $\mu(1 \times M)$ as in Equation (3).

4: Calculate between-class matrix $S_B(M \times M)$ as follows:

$$S_B = \sum_{i=1}^{c} n_i(\mu_i - \mu)(\mu_i - \mu)^T \tag{11}$$

5: Compute within-class matrix $S_W(M \times M)$, as follows:

$$S_W = \sum_{j=1}^{c} \sum_{i=1}^{n_j} (x_{ij} - \mu_j)(x_{ij} - \mu_j)^T \tag{12}$$

where $x_{ij}$ represents the $i^{th}$ sample in the $j^{th}$ class.

6: From Equation (11 and 12), the matrix $W$ that maximizing Fisher's formula which is defined in Equation (7) is calculated as follows, $W = S_W^{-1} S_B$. The eigenvalues ($\lambda$) and eigenvectors ($V$) of $W$ are then calculated.

7: Sorting eigenvectors in descending order according to their corresponding eigenvalues. The first $k$ eigenvectors are then used as a lower dimensional space ($V_k$).

8: Project all original samples ($X$) onto the lower dimensional space of LDA as in Equation (9).

**Algorithm 2.** : Linear Discriminant Analysis (LDA): Class-Dependent

1: Given a set of $N$ samples $[x_i]_{i=1}^{N}$, each of which is represented as a row of length $M$ as in Fig. (1, step(**A**)), and $X(N \times M)$ is given by,

$$X = \begin{bmatrix} x_{(1,1)} & x_{(1,2)} & \cdots & x_{(1,M)} \\ x_{(2,1)} & x_{(2,2)} & \cdots & x_{(2,M)} \\ \vdots & \vdots & \vdots & \vdots \\ x_{(N,1)} & x_{(N,2)} & \cdots & x_{(N,M)} \end{bmatrix} \tag{13}$$

2: Compute the mean of each class $\mu_i(1 \times M)$ as in Equation (2).

3: Compute the total mean of all data $\mu(1 \times M)$ as in Equation (3).

4: Calculate between-class matrix $S_B(M \times M)$ as in Equation (11)

5: **for all** Class $i, i = 1, 2, \ldots, c$ **do**

6: Compute within-class matrix of each class $S_{W_i}(M \times M)$, as follows:

$$S_{W_j} = \sum_{x_i \in \omega_j} (x_i - \mu_j)(x_i - \mu_j)^T \tag{14}$$

7: Construct a transformation matrix for each class ($W_i$) as follows:

$$W_i = S_{W_i}^{-1} S_B \tag{15}$$

8: The eigenvalues ($\lambda^i$) and eigenvectors ($V^i$) of each transformation matrix ($W_i$) are then calculated, where $\lambda^i$ and $V^i$ represent the calculated eigenvalues and eigenvectors of the $i^{th}$ class, respectively.

9: Sorting the eigenvectors in descending order according to their corresponding eigenvalues. The first $k$ eigenvectors are then used to construct a lower dimensional space for each class $V_k^i$.

10: Project the samples of each class ($\omega_i$) onto their lower dimensional space ($V_k^i$), as follows:

$$\Omega_j = x_i V_k^j,\ x_i \in \omega_j \tag{16}$$

where $\Omega_j$ represents the projected samples of the class $\omega_j$.

11: **end for**

each class which increases the complexity of the class-dependent algorithm. Totally, the computational complexity of the class-dependent algorithm is $O(NM^2)$ if $N > M$; otherwise, the complexity is $O(cM^3)$. Hence, the class-dependent method needs computations more than class-independent method.

In our case, we assumed that there are 40 classes and each class has ten samples. Each sample is represented by 4096 features ($M > N$). Thus, the computational complexity of the class-independent method is $O(M^3) = 4096^3$, while the class-dependent method needs $O(cM^3) = 40 \times 4096^3$.

## 3. Numerical Examples

In this section, two numerical examples will be presented. The two numerical examples explain the steps to calculate the LDA space and how the LDA technique is used to discriminate between only two different classes. In the first example, the lower-dimensional space is calculated using the class-independent method, while in the second example, the class-dependent method is used. Moreover, a comparison between the lower dimensional spaces of each method is presented. In all numerical examples, the numbers are rounded up to the nearest hundredths (i.e. only two digits after the decimal point are displayed).

The first four steps of both class-independent and class-dependent methods are common as illustrated in Algorithms (1 and 2). Thus, in this section, we show how these steps are calculated.

Given two different classes, $\omega_1(5 \times 2)$ and $\omega_2(6 \times 2)$ have ($n_1 = 5$) and ($n_2 = 6$) samples, respectively. Each sample in both classes is represented by two features (i.e. $M = 2$) as follows:

$$\omega_1 = \begin{bmatrix} 1.00 & 2.00 \\ 2.00 & 3.00 \\ 3.00 & 3.00 \\ 4.00 & 5.00 \\ 5.00 & 5.00 \end{bmatrix} \text{ and } \omega_2 = \begin{bmatrix} 4.00 & 2.00 \\ 5.00 & 0.00 \\ 5.00 & 2.00 \\ 3.00 & 2.00 \\ 5.00 & 3.00 \\ 6.00 & 3.00 \end{bmatrix} \tag{17}$$

To calculate the lower dimensional space using LDA, first the mean of each class $\mu_j$ is calculated. The total mean $\mu(1 \times 2)$ is then calculated, which represents the mean of all means of all classes. The values of the mean of each class and the total mean are shown below,

$$\begin{aligned} \mu_1 = \begin{bmatrix} 3.00 & 3.60 \end{bmatrix},\ \mu_2 = \begin{bmatrix} 4.67 & 2.00 \end{bmatrix}, \text{ and} \\ \mu = \begin{bmatrix} \frac{5}{11}\mu_1 & \frac{6}{11}\mu_2 \end{bmatrix} = \begin{bmatrix} 3.91 & 2.727 \end{bmatrix} \end{aligned} \tag{18}$$

The between-class variance of each class ($S_{B_i}(2\times 2)$) and the total between-class variance ($S_B(2\times 2)$ are calculated. The values of the between-class variance of the first class ($S_{B_1}$) is equal to,

$$S_{B_1} = n_1(\mu_1-\mu)^T(\mu_1-\mu) = 5[-0.91\ \ 0.87]^T[-0.91\ \ 0.87] = \begin{bmatrix} 4.13 & -3.97 \\ -3.97 & 3.81 \end{bmatrix} \tag{19}$$

Similarly, $S_{B_2}$ is calculated as follows:

$$S_{B_2} = \begin{bmatrix} 3.44 & -3.31 \\ -3.31 & 3.17 \end{bmatrix} \tag{20}$$

The total between-class variance is calculated a follows:

$$S_B = S_{B_1} + S_{B_2} = \begin{bmatrix} 4.13 & -3.97 \\ -3.97 & 3.81 \end{bmatrix} + \begin{bmatrix} 3.44 & -3.31 \\ -3.31 & 3.17 \end{bmatrix} = \begin{bmatrix} 7.58 & -7.27 \\ -7.27 & 6.98 \end{bmatrix} \tag{21}$$

To calculate the within-class matrix, first subtract the mean of each class from each sample in that class and this step is called mean-centering data and it is calculated as follows, $d_i = \omega_i - \mu_i$, where $d_i$ represents centering data of the class $\omega_i$. The values of $d_1$ and $d_2$ are as follows:

$$d_1 = \begin{bmatrix} -2.00 & -1.60 \\ -1.00 & -0.60 \\ 0.00 & -0.60 \\ 1.00 & 1.40 \\ 2.00 & 1.40 \end{bmatrix} \text{ and } d_2 = \begin{bmatrix} -0.67 & 0.00 \\ 0.33 & -2.00 \\ 0.33 & 0.00 \\ -1.67 & 0.00 \\ 0.33 & 1.00 \\ 1.33 & 1.00 \end{bmatrix} \tag{22}$$

In the next two subsections, two different methods are used to calculate the LDA space.

### *3.1. Class-Independent Method*

In this section, the LDA space is calculated using the class-independent method. This method represents the standard method of LDA as in Algorithm (1).

After centring the data, the within-class variance for each class ($S_{W_i}(2\times 2)$) is calculated as follows, $S_{W_j} = d_j^T * d_j = \sum_{i=1}^{n_j}(x_{ij}-\mu_j)^T(x_{ij}-\mu_j)$, where $x_{ij}$ represents the $i^{th}$ sample in the $j^{th}$ class. The total within-class matrix ($S_W(2\times 2)$) is then calculated as follows, $S_W = \sum_{i=1}^{c} S_{W_i}$. The values of the within-class matrix for each class and the total within-class matrix are as follows:

$$S_{W_1} = \begin{bmatrix} 10.00 & 8.00 \\ 8.00 & 7.20 \end{bmatrix},\ S_{W_2} = \begin{bmatrix} 5.33 & 1.00 \\ 1.00 & 6.00 \end{bmatrix},\ S_W = \begin{bmatrix} 15.33 & 9.00 \\ 9.00 & 13.20 \end{bmatrix} \tag{23}$$

The transformation matrix ($W$) in the class-independent method can be obtained as follows, $W = S_W^{-1}S_B$, and the values of ($S_W^{-1}$) and ($W$) are as follows:

$$S_W^{-1} = \begin{bmatrix} 0.11 & -0.07 \\ -0.07 & 0.13 \end{bmatrix} \text{ and } W = \begin{bmatrix} 1.36 & -1.31 \\ -1.48 & 1.42 \end{bmatrix} \tag{24}$$

The eigenvalues ($\lambda(2\times 2)$) and eigenvectors ($V(2\times 2)$) of $W$ are then calculated as follows:

$$\lambda = \begin{bmatrix} 0.00 & 0.00 \\ 0.00 & 2.78 \end{bmatrix} \text{ and } V = \begin{bmatrix} -0.69 & 0.68 \\ -0.72 & -0.74 \end{bmatrix} \tag{25}$$

From the above results it can be noticed that, the second eigenvector ($V_2$) has corresponding eigenvalue more than the first one ($V_1$), which reflects that, the second eigenvector is more robust than the first one; hence, it is selected to construct the lower dimensional space. The original data is projected on the lower dimensional space, as follows, $y_i = \omega_i V_2$, where $y_i(n_i\times 1)$ represents the data after projection of the $i^{th}$ class, and its values will be as follows:

$$y_1 = \omega_1 V_2 = \begin{bmatrix} 1.00 & 2.00 \\ 2.00 & 3.00 \\ 3.00 & 3.00 \\ 4.00 & 5.00 \\ 5.00 & 5.00 \end{bmatrix} \begin{bmatrix} 0.68 \\ -0.74 \end{bmatrix} = \begin{bmatrix} -0.79 \\ -0.85 \\ -0.18 \\ -0.97 \\ -0.29 \end{bmatrix} \tag{26}$$

Similarly, $y_2$ is as follows:

$$y_2 = \omega_2 V_2 = \begin{bmatrix} 1.24 \\ 3.39 \\ 1.92 \\ 0.56 \\ 1.18 \\ 1.86 \end{bmatrix} \tag{27}$$

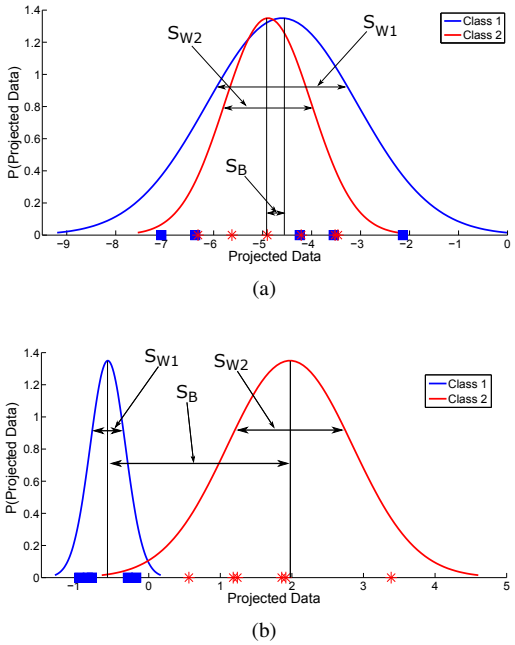



Fig. 4. Probability density function of the projected data of the first example, (a) the projected data on $V_1$, (b) the projected data on $V_2$.

Figure (4) illustrates a *probability density function* (pdf) graph of the projected data ($y_i$) on the two eigenvectors ($V_1$ and $V_2$). A comparison of the two eigenvectors reveals the following :

- The data of each class is completely discriminated when it is projected on the second eigenvector (see Fig. (4)**(b)**) than the first one (see Fig. (4**a**)). In other words, the second eigenvector maximizes the between-class variance more than the first one.
- The within-class variance (i.e. the variance between the same class samples) of the two classes are minimized when the data are projected on the second eigenvector. As shown in Fig. ((4)**(b)**), the within-class variance of the first class is small compared with Fig. ((4)**(a)**).

### 3.2. *Class-Dependent Method*

In this section, the LDA space is calculated using the class-dependent method. As mentioned in Sect. 2.5, the class-dependent method aims to calculate a separate transformation matrix ($W_i$) for each class.

The within-class variance for each class ($S_{W_i}(2 \times 2)$) is calculated as in class-independent method. The transformation matrix ($W_i$) for each class is then calculated as follows, $W_i = S_{W_i}^{-1} S_B$. The values of the two transformation matrices ($W_1$ and $W_2$) will be as follows:

$$W_1 = S_{W_1}^{-1} S_B = \begin{bmatrix} 10.00 & 8.00 \\ 8.00 & 7.20 \end{bmatrix}^{-1} \begin{bmatrix} 7.58 & -7.27 \\ -7.27 & 6.98 \end{bmatrix} = \begin{bmatrix} 0.90 & -1.00 \\ -1.00 & 1.25 \end{bmatrix} \begin{bmatrix} 7.58 & -7.27 \\ -7.27 & 6.98 \end{bmatrix} = \begin{bmatrix} 14.09 & -13.53 \\ -16.67 & 16.00 \end{bmatrix} \tag{28}$$

Similarly, $W_2$ is calculated as follows:

$$W_2 = \begin{bmatrix} 1.70 & -1.63 \\ -1.50 & 1.44 \end{bmatrix} \tag{29}$$

The eigenvalues ($\lambda_i$) and eigenvectors ($V_i$) for each transformation matrix ($W_i$) are calculated, and the values of the eigenvalues and eigenvectors are shown below.

$$\lambda_{\omega_1} = \begin{bmatrix} 0.00 & 0.00 \\ 0.00 & 30.01 \end{bmatrix} \text{ and } V_{\omega_1} = \begin{bmatrix} -0.69 & 0.65 \\ -0.72 & -0.76 \end{bmatrix} \tag{30}$$

$$\lambda_{\omega_2} = \begin{bmatrix} 3.14 & 0.00 \\ 0.00 & 0.00 \end{bmatrix} \text{ and } V_{\omega_2} = \begin{bmatrix} 0.75 & 0.69 \\ -0.66 & 0.72 \end{bmatrix} \tag{31}$$

where $\lambda_{\omega_i}$ and $V_{\omega_i}$ represent the eigenvalues and eigenvectors of the $i^{th}$ class, respectively.

From the results shown (above) it can be seen that, the second eigenvector of the first class ($V_{\omega_1}^{\{2\}}$) has corresponding eigenvalue more than the first one; thus, the second eigenvector is used as a lower dimensional space for the first class as follows, $y_1 = \omega_1 * V_{\omega_1}^{\{2\}}$, where $y_1$ represents the projection of the samples of the first class. While, the first eigenvector in the second class ($V_{\omega_2}^{\{1\}}$) has corresponding eigenvalue more than the second one. Thus, $V_{\omega_2}^{\{1\}}$ is used to project the data of the second class as follows, $y_2 = \omega_2 * V_{\omega_2}^{\{1\}}$, where $y_2$ represents the projection of the samples of the second class. The values of $y_1$ and $y_2$ will be as follows:

$$y_1 = \begin{bmatrix} -0.88 \\ -1.00 \\ -0.35 \\ -1.24 \\ -0.59 \end{bmatrix} \text{ and } y_2 = \begin{bmatrix} 1.68 \\ 3.76 \\ 2.43 \\ 0.93 \\ 1.77 \\ 2.53 \end{bmatrix} \tag{32}$$

Figure (5) shows a pdf graph of the projected data (i.e. $y_1$ and $y_2$) on the two eigenvectors ($V_{\omega_1}^{\{2\}}$ and $V_{\omega_2}^{\{1\}}$) and a number of findings are revealed the following:

- First, the projection data of the two classes are efficiently discriminated.
- Second, the within-class variance of the projected samples is lower than the within-class variance of the original samples.

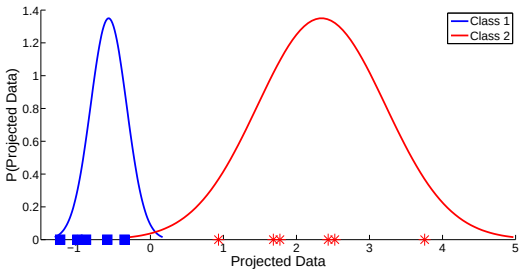



Fig. 5. Probability density function (pdf) of the projected data using class-dependent method, the first class is projected on $V_{\omega_1}^{\{2\}}$, while the second class is projected on $V_{\omega_2}^{\{1\}}$.

### 3.3. Discussion

In these two numerical examples, the LDA space is calculated using class-dependent and class-independent methods.

Figure (6) shows a further explanation of the two methods as following:

- Class-Independent: As shown from the figure, there are two eigenvectors, $V_1$ (dotted black line) and $V_2$ (solid black line). The differences between the two eigenvectors are as follows:
  - The projected data on the second eigenvector ($V_2$) which has the highest corresponding eigenvalue will discriminate the data of the two classes better than the first eigenvector. As shown in the figure, the distance between the projected means $m_1 - m_2$ which represents $S_B$, increased when the data are projected on $V_2$ than $V_1$.
  - The second eigenvector decreases the within-class variance much better than the first eigenvector. Figure (6) illustrates that the within-class variance of the first class ($S_{W_1}$) was much smaller when it was projected on $V_2$ than $V_1$.
  - As a result of the above two findings, $V_2$ is used to construct the LDA space.
- Class-Dependent: As shown from the figure, there are two eigenvectors, $V_{\omega_1}^{\{2\}}$ (red line) and $V_{\omega_2}^{\{1\}}$ (blue line), which represent the first and second classes, respectively. The differences between the two eigenvectors are as following:
  - Projecting the original data on the two eigenvectors discriminates between the two classes. As shown in the figure, the distance between the projected means $m_1 - m_2$ is larger than the distance between the original means $\mu_1 - \mu_2$.
  - The within-class variance of each class is decreased. For example, the within-class variance of the first class ($S_{W_1}$) is decreased when it is projected on its corresponding eigenvector.
  - As a result of the above two findings, $V_{\omega_1}^{\{2\}}$ and $V_{\omega_2}^{\{1\}}$ are used to construct the LDA space.
- Class-Dependent vs. Class-Independent: The two LDA methods are used to calculate the LDA space, but a class-dependent method calculates separate lower dimensional spaces for each class which has two main limitations: (1) it needs more CPU time and calculations more than class-independent method; (2) it may lead to *SSS* problem because the number of samples in each class affects the singularity of ${S_{W_i}}^2$.

These findings reveal that the standard LDA technique used the class-independent method rather than using the class-dependent method.

## 4. Main Problems of LDA

Although LDA is one of the most common data reduction techniques, it suffers from two main problems: the *Small Sample Size* (SSS) and linearity problems. In the next two subsections, these two problems will be explained, and some of the state-of-the-art solutions are highlighted.

### 4.1. Linearity problem

LDA technique is used to find a linear transformation that discriminates between different classes. However, if the classes are non-linearly separable, LDA can not find a lower dimensional space. In other words, LDA fails to find the LDA space when the discriminatory information are not in the means of classes. Fig-

[2] *SSS* problem will be explained in Sect. 4.2

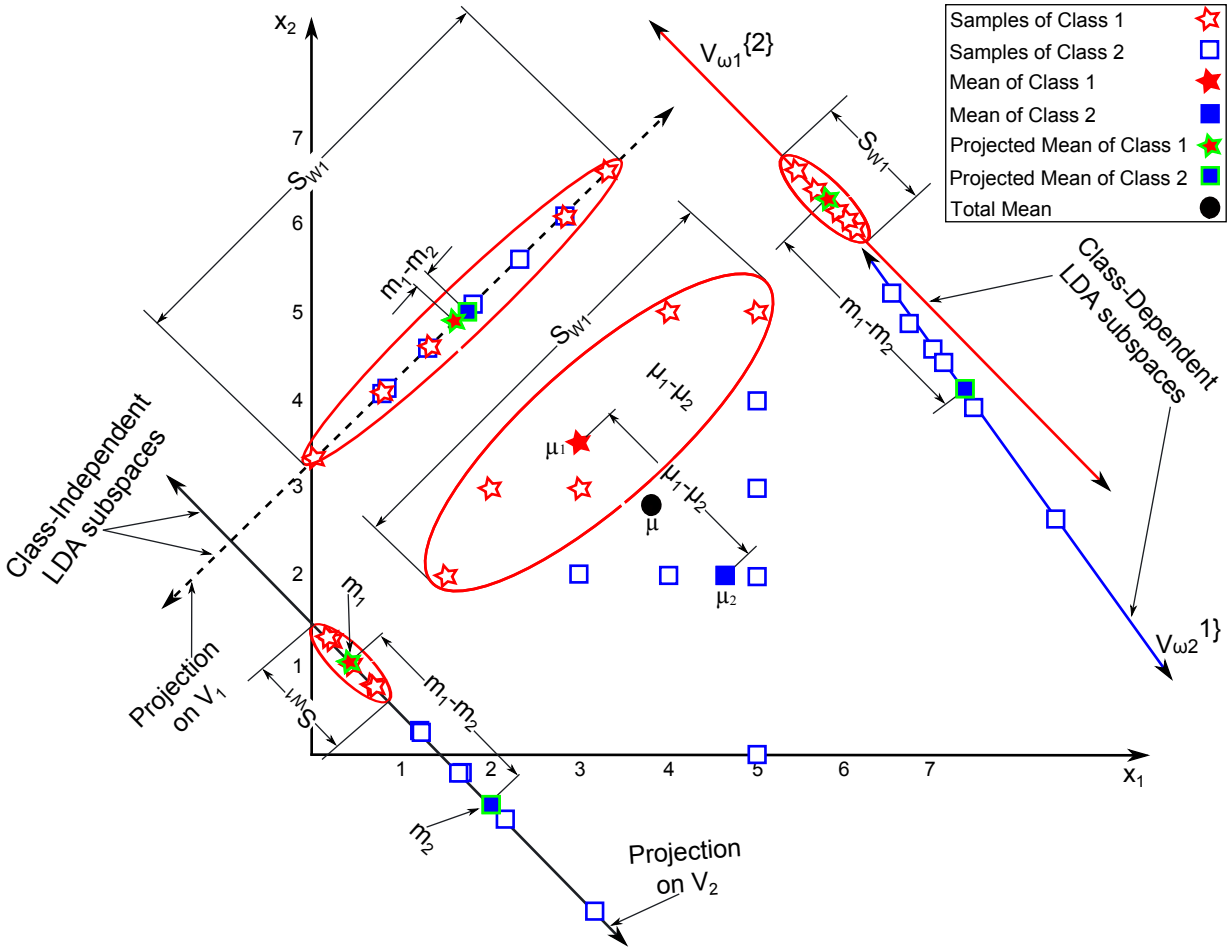



Fig. 6. Illustration of the example of the two different methods of LDA methods. The blue and red lines represent the first and second eigenvectors of the class-dependent approach, respectively, while the solid and dotted black lines represent the second and first eigenvectors of class-independent approach, respectively.

ure (7) shows how the discriminatory information does not exist in the mean, but in the variance of the data. This is because the means of the two classes are equal. The mathematical interpretation for this problem is as follows: if the means of the classes are approximately equal, so the $S_B$ and $W$ will be zero. Hence, the LDA space cannot be calculated.

One of the solutions of this problem is based on the transformation concept, which is known as *a kernel methods or functions* [50,3]. Figure (7) illustrates how the transformation is used to map the original data into a higher dimensional space; hence, the data will be linearly separable, and the LDA technique can find the lower dimensional space in the new space. Figure (8) graphically and mathematically shows how two non-separable classes in one-dimensional space are transformed into a two-dimensional space (i.e. higher dimensional space); thus, allowing linear separation.

The kernel idea is applied in Support Vector Machine (SVM) [49,66,69] Support Vector Regression (SVR)[58], PCA [51], and LDA [50]. Let $\phi$ represents a nonlinear mapping to the new feature space $\mathcal{Z}$. The transformation matrix ($W$) in the new feature space ($\mathcal{Z}$) is calculated as in Equation (33).

$$F(W) = \max \left| \frac{W^T S_B^{\phi} W}{W^T S_W^{\phi} W} \right| \tag{33}$$

where $W$ is a transformation matrix and $\mathcal{Z}$ is the new feature space. The between-class matrix ($S_B^{\phi}$) and the within-class matrix ($S_W^{\phi}$) are defined as follows:

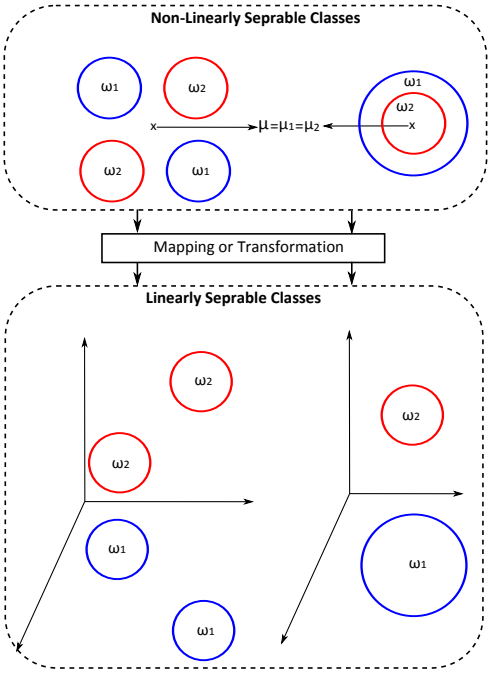



Fig. 7. Two examples of two non-linearly separable classes, top panel shows how the two classes are non-separable, while the bottom shows how the transformation solves this problem and the two classes are linearly separable.

$$S_B^{\phi} = \sum_{i=1}^{c} n_i (\mu_i^{\phi} - \mu^{\phi})(\mu_i^{\phi} - \mu^{\phi})^T \tag{34}$$

$$S_W^{\phi} = \sum_{j=1}^{c} \sum_{i=1}^{n_j} (\phi\{x_{ij}\} - \mu_j^{\phi})(\phi\{x_{ij}\} - \mu_j^{\phi})^T \tag{35}$$

where $\mu_i^{\phi} = \frac{1}{n_i}\sum_{i=1}^{n_i} \phi\{x_i\}$ and $\mu^{\phi} = \frac{1}{N}\sum_{i=1}^{N} \phi\{x_i\} = \sum_{i=1}^{c} \frac{n_i}{N}\mu_i^{\phi}$

Thus, in kernel LDA, all samples are transformed non-linearly into a new space $\mathcal{Z}$ using the function $\phi$. In other words, the $\phi$ function is used to map the original features into $\mathcal{Z}$ space by creating a nonlinear combination of the original samples using a dot-products of it [3]. There are many types of kernel functions to achieve this aim. Examples of these function include Gaussian or Radial Basis Function (RBF), $K(x_i,x_j) = exp(-||x_i - x_j||^2/2\sigma^2)$, where $\sigma$ is a positive parameter, and the polynomial kernel of degree $d$, $K(x_i,x_j) = (\langle x_i,x_j\rangle + c)^d$, [3,51,72,29].

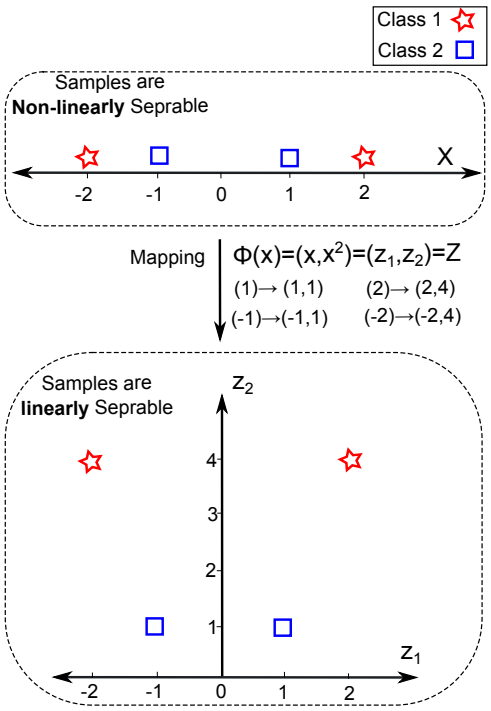



Fig. 8. Example of kernel functions, the samples lie on the top panel ($X$) which are represented by a line (i.e. one-dimensional space) are non-linearly separable, where the samples lie on the bottom panel ($Z$) which are generated from mapping the samples of the top space are linearly separable.

### *4.2. Small Sample Size Problem*

#### *4.2.1. Problem Definition*

Singularity, Small Sample Size (SSS), or under-sampled problem is one of the big problems of LDA technique. This problem results from high-dimensional pattern classification tasks or a low number of training samples available for each class compared with the dimensionality of the sample space [30,38,85,82].

The *SSS* problem occurs when the $S_W$ is singular[3]. The upper bound of the rank[4] of $S_W$ is $N-c$, while the dimension of $S_W$ is $M \times M$ [38,17]. Thus, in most cases $M >> N-c$ which leads to *SSS* problem. For example, in face recognition applications, the size of the face image my reach to $100 \times 100 = 10000$ pixels, which represent high-dimensional features and it leads to a singularity problem.

[3] A matrix is singular if it is square, does not have a matrix inverse, the determinant is zeros; hence, not all columns and rows are independent

[4] The rank of the matrix represents the number of linearly independent rows or columns

*4.2.2. Common Solutions to SSS Problem:*

There are many studies that proposed many solutions for this problem; each has its advantages and drawbacks.

- **Regularization (RLDA):** In regularization method, the identity matrix is scaled by multiplying it by a regularization parameter ($\eta > 0$) and adding it to the within-class matrix to make it non-singular [18,38,82,45]. Thus, the diagonal components of the within-class matrix are biased as follows, $S_W = S_W + \eta I$. However, choosing the value of the regularization parameter requires more tuning and a poor choice for this parameter can degrade the performance of the method [38,45]. Another problem of this method is that the parameter $\eta$ is just added to perform the inverse of $S_W$ and has no clear mathematical interpretation [38,57].
- **Sub-space:** In this method, a non-singular intermediate space is obtained to reduce the dimension of the original data to be equal to the rank of $S_W$; hence, $S_W$ becomes full-rank[5], and then $S_W$ can be inverted. For example, Belhumeur et al. [4] used PCA, to reduce the dimensions of the original space to be equal to $N-c$ (i.e. the upper bound of the rank of $S_W$). However, as reported in [22], losing some discriminant information is a common drawback associated with the use of this method.
- **Null Space:** There are many studies proposed to remove the null space of $S_W$ to make $S_W$ full-rank; hence, invertible. The drawback of this method is that more discriminant information is lost when the null space of $S_W$ is removed, which has a negative impact on how the lower dimensional space satisfies the LDA goal [83].

Four different variants of the LDA technique that are used to solve the SSS problem are introduced as follows:

*PCA + LDA technique: In this technique, the original d-dimensional features are first reduced to h-dimensional feature space using PCA, and then the LDA is used to further reduce the features to k-dimensions. The PCA is used in this technique to reduce the dimensions to make the rank of $S_W$ is $N-c$ as reported in [4]; hence, the SSS problem is addressed. However, the PCA neglects some discriminant information, which may reduce the classification performance [57,60].*

*Direct LDA technique: Direct LDA (DLDA) is one of the well-known techniques that are used to solve the SSS problem. This technique has two main steps [83]. In the first step, the transformation matrix, $W$, is computed to transform the training data to the range space of $S_B$. In the second step, the dimensionality of the transformed data is further transformed using some regulating matrices as in Algorithm 4. The benefit of the DLDA is that there is no discriminative features are neglected as in PCA+LDA technique [83].*

*Regularized LDA technique:* In the Regularized LDA (RLDA), a small perturbation is add to the $S_W$ matrix to make it non-singular as mentioned in [18]. This regularization can be applied as follows:

$$(S_W + \eta I)^{-1} S_B w_i = \lambda_i w_i \tag{36}$$

where $\eta$ represents a regularization parameter. The diagonal components of the $S_W$ are biased by adding this small perturbation [18,13]. However, the regularization parameter need to be tuned and poor choice of it can degrade the generalization performance [57].

*Null LDA technique: The aim of the NLDA technique is to find the orientation matrix $W$, and this can be achieved using two steps. In the first step, the range space of the $S_W$ is neglected, and the data are projected only on the null space of $S_W$ as follows, $S_W W = 0$. In the second step, the aim is to search for $W$ that satisfies $S_B W = 0$ and maximizes $|W^T S_B W|$. The higher dimensionality of the feature space may lead to computational problems. This problem can be solved by (1) using the PCA technique as a pre-processing step, i.e. before applying the NLDA technique, to reduce the dimension of feature space to be $N-1$; by removing the null space of $S_T = S_B + S_W$ [57], (2) using the PCA technique before the second step of the NLDA technique [54]. Mathematically, in the Null LDA (NLDA) technique, the h column vectors of the transformation matrix $W = [w_1, w_2, \ldots, w_h]$ are taken to be the null space of the $S_W$ as follows, $w_i^T S_W w_i = 0$, $\forall i = 1 \ldots h$, where $w_i^T S_B w_i \neq 0$. Hence, $M-(N-c)$ linearly independent vectors are used to form a new orientation matrix, which is used to maximize $|W^T S_B W|$ subject to the constraint $|W^T S_W W| = 0$ as in Equation (37).*

$$W = arg \max_{|W^T S_W W|=0} |W^T S_B W| \tag{37}$$

[5] $A$ is a full-rank matrix if all columns and rows of the matrix are independent, (i.e. rank($A$)= # rows= #cols) [23]

## 5. Applications of the LDA technique

In many applications, due to the high number of features or dimensionality, the LDA technique have been used. Some of the applications of the LDA technique and its variants are described as follows:

### *5.1. Biometrics Applications*

Biometrics systems have two main steps, namely, feature extraction (including pre-processing steps) and recognition. In the first step, the features are extracted from the collected data, e.g. face images, and in the second step, the unknown samples, e.g. unknown face image, is identified/verified. The LDA technique and its variants have been applied in this application. For example, in [83,10,41,75,20,68], the LDA technique have been applied on face recognition. Moreover, the LDA technique was used in Ear [84], fingerprint [44], gait [5], and speech [24] applications. In addition, the LDA technique was used with animal biometrics as in [65,19].

### *5.2. Agriculture Applications*

In agriculture applications, an unknown sample can be classified into a pre-defined species using computational models [64]. In this application, different variants of the LDA technique was used to reduce the dimension of the collected features as in [9,26,46,21,64, 63].

### *5.3. Medical Applications*

In medical applications, the data such as the DNA microarray data consists of a large number of features or dimensions. Due to this high dimensionality, the computational models need more time to train their models, which may be infeasible and expensive. Moreover, this high dimensionality reduces the classification performance of the computational model and increases its complexity. This problem can be solved using LDA technique to construct a new set of features from a large number of original features. There are many papers have been used LDA in medical applications [54,52,53,16,39,55,8].

## 6. Packages

In this section, some of the available packages that are used to compute the space of LDA variants. For example, WEKA[6] is a well-known Java-based data mining tool with open source machine learning software such as classification, association rules, regression, pre-processing, clustering, and visualization. In WEKA, the machine learning algorithms can be applied directly on the dataset or called from person's Java code. XLSTAT[7] is another data analysis and statistical package for Microsoft Excel that has a wide variety of dimensionality reduction algorithms including LDA. dChip[8] package is also used for visualization of gene expression and SNP microarray including some data analysis algorithms such as LDA, clustering, and PCA. LDA-SSS[9] is a Matlab package, and it contains several algorithms related to the LDA techniques and its variants such as DLDA, PCA+LDA, and NLDA. MASS[10] package is based on R, and it has functions that are used to perform linear and quadratic discriminant function analysis. Dimensionality reduction[11] package is mainly written in Matlab, and it has a number of dimensionality reduction techniques such as ULDA, QLDA, and KDA. DTREG[12] is a software package that is used for medical data and modeling business, and it has several predictive modeling methods such as LDA, PCA, linear regression, and decision trees.

## 7. Experimental Results and Discussion

In this section, two experiments were conducted to illustrate: (1) how the LDA is used for different applications, (2) what is the relation between its parameter (Eigenvectors) and the accuracy of a classification problem, (3) when the *SSS* problem could appear and a method for solving it.

[6] http:/www.cs.waikato.ac.nz/ml/weka/
[7] http://www.xlstat.com/en/
[8] https://sites.google.com/site/dchipsoft/home
[9] http://www.staff.usp.ac.fj/sharma_al/index.htm
[10] http://www.statmethods.net/advstats/discriminant.html
[11] http://www.public.asu.edu/*jye02/Software/index.html
[12] http://www.dtreg.com/index.htm

### 7.1. Experimental Setup

This section gives an overview of the databases, the platform, and the machine specification used to conduct our experiments. Different biometric datasets were used in the experiments to show how the LDA using its parameter behaves with different data. These datasets are described as follows:

- ORL dataset[13] face images dataset (Olivetti Research Laboratory, Cambridge) [48], which consists of 40 distinct individuals, was used. In this dataset, each individual has ten images taken at different times and varying light conditions. The size of each image is $92 \times 112$.
- Yale dataset[14] is another face images dataset which contains 165 grey scale images in GIF format of 15 individuals [78]. Each individual has 11 images in different expressions and configuration: center-light, happy, left-light, with glasses, normal, right-light, sad, sleepy, surprised, and a wink.
- 2D ear dataset (Carreira-Perpinan,1995)[15] images dataset [7] was used. The ear data set consists of 17 distinct individuals. Six views of the left profile from each subject were taken under a uniform, diffuse lighting.

In all experiments, $k$-fold cross-validation tests have used. In $k$-fold cross-validation, the original samples of the dataset were randomly partitioned into $k$ subsets of (approximately) equal size and the experiment is run $k$ times. For each time, one subset was used as the testing set and the other $k-1$ subsets were used as the training set. The average of the $k$ results from the folds can then be calculated to produce a single estimation. In this study, the value of $k$ was set to 10.

The images in all datasets resized to be $64 \times 64$ and $32 \times 32$ as shown in Table (2). Figure (9) shows samples of the used datasets and Table (2) shows a description of the datasets used in our experiments.

In all experiments, to show the effect of the LDA with its eigenvector parameter and its *SSS* problem on the classification accuracy, the Nearest Neighbour classifier was used. This classifier aims to classify the testing image by comparing its position in the LDA space with the positions of training images. Furthermore, class-independent LDA was used in all experiments.

[13] http://www.cam-orl.co.uk

[14] http://vision.ucsd.edu/content/yale-face-database

[15] http://faculty.ucmerced.edu/mcarreira-perpinan/software.html

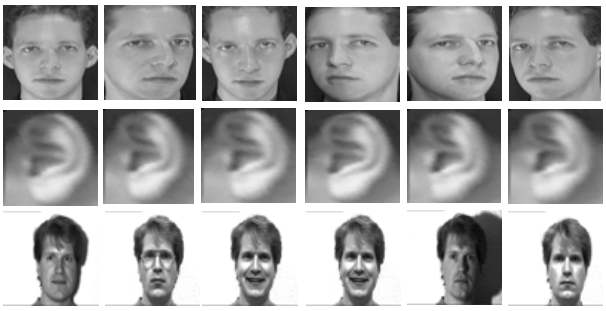


Fig. 9. Samples of the first individual in: ORL face dataset (top row); Ear dataset (middle row), and Yale face dataset (bottom row).

Table 2
Dataset description.

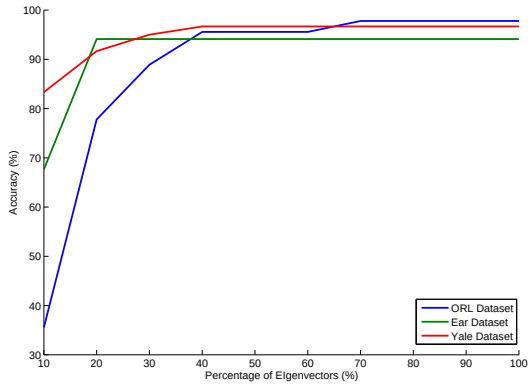
| **Dataset** | **Dimension ($M$)** | **No. of Samples ($N$)** | **No. of classes ($c$)** |
|---|---|---|---|
| $ORL_{64\times64}$ | 4096 | 400 | 40 |
| $ORl_{32\times32}$ | 1024 | | |
| $Ear_{64\times64}$ | 4096 | 102 | 17 |
| $Ear_{32\times32}$ | 1024 | | |
| $Yale_{64\times64}$ | 4096 | 165 | 15 |
| $Yale_{32\times32}$ | 1024 | | |

Moreover, Matlab Platform (R2013b) and using a PC with the following specifications: Intel(R) Core(TM) i5-2400 CPU @ 3.10 GHz and 4.00 GB RAM, under Windows 32-bit operating system were used in our experiments.

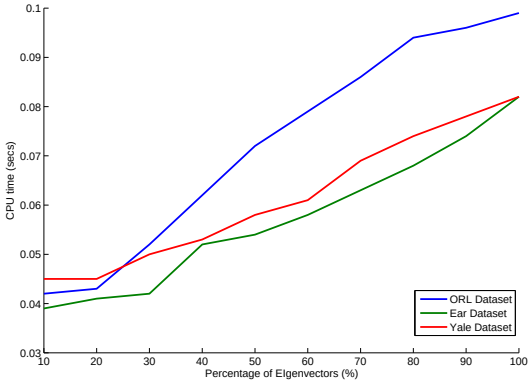
### 7.2. Experiment on LDA Parameter (Eigenvectors)

The aim of this experiment is to investigate the relation between the number of eigenvectors used in the LDA space, and the classification accuracy based on these eigenvectors and the required CPU time for this classification.

As explained earlier that the LDA space consists of $k$ eigenvectors, which are sorted according to their robustness (i.e. their eigenvalues). The robustness of each eigenvector reflects its ability to discriminate between different classes. Thus, in this experiment, it will be checked whether increasing the number of eigenvectors would increase the total robustness of the constructed LDA space; hence, different classes could be well discriminated. Also, it will be tested whether increasing the number of eigenvectors would increase the dimension of the LDA space and the projected data; hence, CPU time increases. To investi-




(a)




(b)

Fig. 10. Accuracy and CPU time of the LDA techniques using different percentages of eigenvectors, (a) Accuracy (b) CPU time.

gate these issue, three datasets listed in Table (2) (i.e. $ORL_{32\times32}$, $Ear_{32\times32}$, $Yale_{32\times32}$), were used. Moreover, seven, four, and eight images from each subject in the ORL, ear, Yale datasets, respectively, are used in this experiment. The results of this experiment are presented in Fig. (10).

From Fig. (10) it can be noticed that the accuracy and CPU time are proportional with the number of eigenvectors which are used to construct the LDA space. Thus, the choice of using LDA in a specific application should consider a trade-off between these factors. Moreover, from Fig. (10a), it can be remarked that when the number of eigenvectors used in computing the LDA space was increased, the classification accuracy was also increased to a specific extent after which the accuracy remains constant. As seen in Fig. (10a), this extent differs from application to another. For example, the accuracy of the ear dataset remains constant when the percentage of the used eigenvectors is more than 10%. This was expected as the eigenvectors of the LDA space are sorted according to their robustness (see Sect. 2.6). Similarly, in ORL and Yale datasets the accuracy became approximately constant when the percentage of the used eigenvectors is more than 40%. In terms of CPU time, Fig. (10b) shows the CPU time using different percentages of the eigenvectors. As shown, the CPU time increased dramatically when the number of eigenvectors increases.

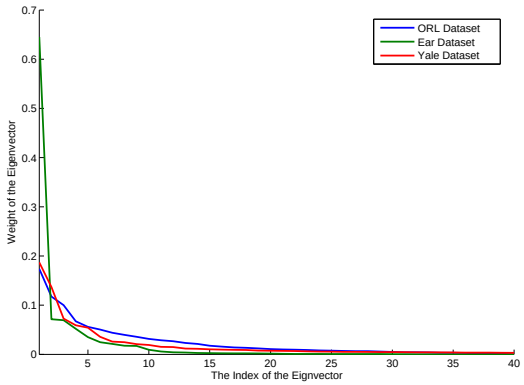



Fig. 11. The robustness of the first 40 eigenvectors of the LDA technique using $ORL_{32\times32}$, $Ear_{32\times32}$, and $Yale_{32\times32}$ datasets.

Fig. (11) shows the weights[16] of the first 40 eigenvectors which confirms our findings. From these results, we can conclude that the high order eigenvectors of the data of each application (the first 10 % of ear database and the first 40 % of ORL and Yale datasets) are robust enough to extract and save the most discriminative features which are used to achieve a good accuracy.

These experiments confirmed that increasing the number of eigenvectors will increase the dimension of the LDA space; hence, CPU time increases. Consequently, the amount of discriminative information and the accuracy increases.

### *7.3. Experiments on the Small Sample Size problem*

The aim of this experiment is to show when the LDA is subject to the *SSS* problem and what are the methods that could be used to solve this problem. In this experiment, PCA-LDA [4] and Direct-LDA [83,22] methods

[16]The weight of the eigenvector represents the ratio of its corresponding eigenvalue ($\lambda_i$) to the total of all eigenvalues ($\lambda_i, i = 1,2,\ldots,k$) as follows, $\frac{\lambda_i}{\sum_{j=1}^{k}\lambda_j}$

are used to address the *SSS* problem. A set of experiments was conducted to show how the two methods interact with the *SSS* problem, including the number of samples in each class, the total number of classes used, and the dimension of each sample.

As explained in Sect. 4, using LDA directly may lead to the *SSS* problem when the dimension of the samples are much higher than the total number of samples. As shown in Table (2), the size of each image or sample of the ORL dataset is $_{64\times64}$ is 4096 and the total number of samples is 400. The mathematical interpretation of this point shows that the dimension of $S_W$ is $M \times M$, while the upper bound of the rank of $S_W$ is $N-c$ [79,82]. Thus, all the datasets which are reported in Table (2) lead to singular $S_W$.

To address this problem, PCA-LDA and direct-LDA methods were implemented. In the PCA-LDA method, PCA is first used to reduce the dimension of the original data to make $S_W$ full-rank, and then standard LDA can be performed safely in the subspace of $S_W$ as in Algorithm (3). For more details of the PCA-LDA method are reported in [4]. In the direct-LDA method, the null-space of $S_W$ matrix is removed to make $S_W$ full-rank, then standard LDA space can be calculated as in Algorithm (4). More details of direct-LDA methods are found in [83].

Table (2) illustrates various scenarios designed to test the effect of different dimensions on the rank of $S_W$ and the accuracy. The results of these scenarios using both PCA-LDA and direct-LDA methods are summarized in Table (3).

As summarized in Table (3), the rank of $S_W$ is very small compared to the whole dimension of $S_W$; hence, the *SSS* problem occurs in all cases. As shown in Table (3), using the PCA-LDA and the Direct-LDA, the *SSS* problem can be solved and the Direct-LDA method achieved results better than PCA-LDA because as reported in [83,22], Direct-LDA method saves approximately all important information for classification while the PCA in PCA-LDA method saves the information with high variance.

**Algorithm 3.** : PCA-LDA.

1: Read the training images ($X = \{x_1, x_2, \ldots, x_N\}$), where $x_i(ro \times co)$ represents the $i^{th}$ training image, $ro$ and $co$ represent the rows (height) and columns (width) of $x_i$, respectively, $N$ represents the total number of training images.
2: Convert all images in vector representation $Z = \{z_1, z_2, \ldots, z_N\}$, where the dimension of $Z$ is $M \times 1, M = ro \times co$.
3: calculate the mean of each class $\mu_i$, total mean of all data $\mu$, between-class matrix $S_B(M \times M)$, and within-class matrix $S_W(M \times M)$ as in Algorithm (1, Step(2-5)).
4: Use the PCA technique to reduce the dimension of $X$ to be equal to or lower than $r$, where $r$ represents the rank of $S_W$, as follows:

$$X_{PCA} = U^T X \tag{38}$$

where, $U \in \mathcal{R}^{M\times r}$ is the lower dimensional space of the PCA and $X_{PCA}$ represents the projected data on the PCA space.
5: Calculate the mean of each class $\mu_i$, total mean of all data $\mu$, between-class matrix $S_B$, and within-class matrix $S_W$ of $X_{PCA}$ as in Algorithm (1, Step(2-5)).
6: Calculate $W$ as in Equation (7) and then calculate the eigenvalues ($\lambda$) and eigenvectors ($V$) of the $W$ matrix.
7: Sorting the eigenvectors in descending order according to their corresponding eigenvalues. The first $k$ eigenvectors are then used as a lower dimensional space ($V_k$).
8: The original samples ($X$) are first projected on the PCA space as in Equation (38). The projection on the LDA space is then calculated as follows:

$$X_{LDA} = V_k^T X_{PCA} = V_k^T U^T X \tag{39}$$

where $X_{LDA}$ represents the final projected on the LDA space.

## 8. Conclusions

In this paper, the definition, mathematics, and implementation of LDA were presented and explained. The paper aimed to give low-level details on how the LDA technique can address the reduction problem by extracting discriminative features based on maximizing the ratio between the between-class variance, $S_B$, and within-class variance, $S_W$, thus discriminating between different classes. To achieve this aim, the paper followed the approach of not only explaining the steps of calculating the $S_B$, and $S_W$ (i.e. the LDA space) but also visualizing these steps with figures and diagrams to make it easy to understand. Moreover, two LDA methods, i.e. class-dependent and class-independent, are explained and two numerical examples were given

Table 3
Accuracy of the PCA-LDA and direct-LDA methods using the datasets listed in Table (2).

| Dataset | Dim ($S_W$) | # Training images | # Testing images | Rank ($S_W$) | Accuracy (%) | |
|---|---|---|---|---|---|---|
| | | | | | PCA-LDA | Direct-LDA |
| $ORL_{32\times32}$ | 1024×1024 | 5 | 5 | 160 | 75.5 | **88.5** |
| | | 7 | 3 | 240 | 75.5 | **97.5** |
| | | 9 | 1 | 340 | 80.39 | **97.5** |
| $Ear_{32\times32}$ | 1024×1024 | 3 | 3 | 34 | 80.39 | **96.08** |
| | | 4 | 2 | 51 | 88.24 | **94.12** |
| | | 5 | 1 | 68 | 100 | 100 |
| $Yale_{32\times32}$ | 1024×1024 | 6 | 5 | 75 | 78.67 | **90.67** |
| | | 8 | 3 | 105 | 84.44 | **97.79** |
| | | 10 | 1 | 135 | 100 | 100 |
| $ORL_{64\times64}$ | 4096×4096 | 5 | 5 | 160 | 72 | **87.5** |
| | | 7 | 3 | 240 | 81.67 | **96.67** |
| | | 9 | 1 | 340 | 82.5 | **97.5** |
| $Ear_{64\times64}$ | 4096×4096 | 3 | 3 | 34 | 74.5 | **96.08** |
| | | 4 | 2 | 51 | 91.18 | **96.08** |
| | | 5 | 1 | 68 | 100 | 100 |
| $Yale_{64\times64}$ | 4096×4096 | 6 | 5 | 75 | 74.67 | **92** |
| | | 8 | 3 | 105 | 95.56 | **97.78** |
| | | 10 | 1 | 135 | 93.33 | **100** |

The bold values indicate that the corresponding methods obtain best performances

and graphically illustrated to explain how the LDA space using the two methods can be constructed. In all examples, the mathematical interpretation of the robustness and the selection of the eigenvectors as well the data projection were detailed and discussed. Also, LDA common problems (e.g. the *SSS* and linearity) were mathematically explained using graphical examples, then their state-of-the-art solutions are highlighted. Moreover, a detailed implementation of LDA applications was presented. Using three standard datasets, a number of experiments were conducted to (1) investigate and explain the relation between the number of eigenvectors and the robustness of the LDA space, (2) to practically show when the SSS problem occurs and how it can be addressed.

## References


[1] S. Balakrishnama and A. Ganapathiraju. Linear discriminant analysis-a brief tutorial. *Institute for Signal and information Processing*, 1998.

[2] E. Barshan, A. Ghodsi, Z. Azimifar, and M. Z. Jahromi. Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds. *Pattern Recognition*, 44(7):1357–1371, 2011.

[3] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural computation*, 12(10):2385–2404, 2000.

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

[5] N. V. Boulgouris and Z. X. Chi. Gait recognition using radon transform and linear discriminant analysis. *IEEE transactions on image processing*, 16(3):731–740, 2007.

[6] M. Bramer. *Principles of data mining*. Springer, Second Edition, 2013.

[7] M. Carreira-Perpinan. Compression neural networks for feature extraction: Application to human recognition from ear images. *MS thesis, Faculty of Informatics, Technical University of Madrid, Spain*, 1995.

[8] H.-P. Chan, D. Wei, M. A. Helvie, B. Sahiner, D. D. Adler, M. M. Goodsitt, and N. Petrick. Computer-aided classification of mammographic masses and normal tissue: linear discriminant analysis in texture feature space. *Physics in medicine and biology*, 40(5):857, 1995.

[9] L. Chen, N. C. Carpita, W.-D. Reiter, R. H. Wilson, C. Jeffries, and M. C. McCann. A rapid method to screen for cell-wall mutants using discriminant analysis of fourier transform infrared spectra. *The Plant Journal*, 16(3):385–392, 1998.

[10] L.-F. Chen, H.-Y. M. Liao, M.-T. Ko, J.-C. Lin, and G.-J. Yu. A new lda-based face recognition system which can solve the

**Algorithm 4.** : Direct-LDA.

1: Read the training images ($X = \{x_1, x_2, \ldots, x_N\}$), where $x_i(ro \times co)$ represents the $i^{th}$ training image, $ro$ and $co$ represent the rows (height) and columns (width) of $x_i$, respectively, $N$ represents the total number of training images.
2: Convert all images in vector representation $Z = \{z_1, z_2, \ldots, z_N\}$, where the dimension of $Z$ is $M \times 1, M = ro \times co$.
3: Calculate the mean of each class $\mu_i$, total mean of all data $\mu$, between-class matrix $S_B(M \times M)$, and within-class matrix $S_W(M \times M)$ of $X$ as in Algorithm (1, Step(2-5)).
4: Find the $k$ eigenvectors of $S_B$ with non-zeros eigenvalues, and denote them as $U = [u_1, u_2, \ldots, u_k]$ (i.e. $U^T S_b U > 0$).
5: Calculate the eigenvalues and eigenvectors of $U^T S_W U$, and then sort the eigenvalues and discard the eigenvectors which have high values. The selected eigenvectors are denoted by $V$; thus, $V$ represents the null space of $S_W$.
6: The final LDA matrix ($\Psi$) consists of the range [17] $S_B$ and the null space of $S_W$ as follows, $\Psi = UV$.
7: The original data are projected on the LDA space as follows, $Y = X\Psi = XUV$.


small sample size problem. *Pattern recognition*, 33(10):1713–1726, 2000.

[11] D. Coomans, D. Massart, and L. Kaufman. Optimization by statistical linear discriminant analysis in analytical chemistry. *Analytica Chimica Acta*, 112(2):97–122, 1979.

[12] J. Cui and Y. Xu. Three dimensional palmprint recognition using linear discriminant analysis method. In *Proceedings of the second International Conference on Innovations in Bio-inspired Computing and Applications (IBICA), 2011*, pages 107–111. IEEE, 2011.

[13] D.-Q. Dai and P. C. Yuen. Face recognition by regularized discriminant analysis. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 37(4):1080–1085, 2007.

[14] D. Donoho and V. Stodden. When does non-negative matrix factorization give a correct decomposition into parts? In *Advances in neural information processing systems 16*, pages 1141–1148. MIT Press, 2004.

[15] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. John Wiley & Sons, Second Edition, 2012.

[16] S. Dudoit, J. Fridlyand, and T. P. Speed. Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American statistical association*, 97(457):77–87, 2002.

[17] T.-t. Feng and G. Wu. A theoretical contribution to the fast implementation of null linear discriminant analysis method using random matrix multiplication with scatter matrices. *arXiv preprint arXiv:1409.2579*, 2014.

[18] J. H. Friedman. Regularized discriminant analysis. *Journal of the American Statistical Association*, 84(405):165–175, 1989.

[19] T. Gaber, A. Tharwat, A. E. Hassanien, and V. Snasel. Biometric cattle identification approach based on webers local descriptor and adaboost classifier. *Computers and Electronics in Agriculture*, 122:55–66, 2016.

[20] T. Gaber, A. Tharwat, A. Ibrahim, V. Snáel, and A. E. Hassanien. Human thermal face recognition based on random linear oracle (rlo) ensembles. In *International Conference on Intelligent Networking and Collaborative Systems*, pages 91–98. IEEE, 2015.

[21] T. Gaber, A. Tharwat, V. Snasel, and A. E. Hassanien. Plant identification: Two dimensional-based vs. one dimensional-based feature extraction methods. In *10th international conference on soft computing models in industrial and environmental applications*, pages 375–385. Springer, 2015.

[22] H. Gao and J. W. Davis. Why direct lda is not equivalent to lda. *Pattern Recognition*, 39(5):1002–1006, 2006.

[23] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. Johns Hopkins University Press, Fourth Edition, 2012.

[24] R. Haeb-Umbach and H. Ney. Linear discriminant analysis for improved large vocabulary continuous speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (1992)*, volume 1, pages 13–16. IEEE, 1992.

[25] T. Hastie and R. Tibshirani. Discriminant analysis by gaussian mixtures. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 155–176, 1996.

[26] K. Héberger, E. Csomós, and L. Simon-Sarkadi. Principal component and linear discriminant analyses of free amino acids and biogenic amines in hungarian wines. *Journal of agricultural and food chemistry*, 51(27):8055–8060, 2003.

[27] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[28] K. Honda and H. Ichihashi. Fuzzy local independent component analysis with external criteria and its application to knowledge discovery in databases. *International journal of approximate reasoning*, 42(3):159–173, 2006.

[29] Q. Hu, L. Zhang, D. Chen, W. Pedrycz, and D. Yu. Gaussian kernel based fuzzy rough sets: Model, uncertainty measures and applications. *International Journal of Approximate Reasoning*, 51(4):453–471, 2010.

[30] R. Huang, Q. Liu, H. Lu, and S. Ma. Solving the small sample size problem of lda. In *Proceedings of 16$^{th}$ International Conference on Pattern Recognition, 2002.*, volume 3, pages 29–32. IEEE, 2002.

[31] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent component analysis*, volume 46. John Wiley & Sons, 2004.

[32] M. Kirby. *Geometric data analysis: an empirical approach to dimensionality reduction and the study of patterns*. John Wiley & Sons, 2000.

[33] D. T. Larose. *Discovering knowledge in data: an introduction to data mining*. John Wiley & Sons, First Edition, 2014.

[34] T. Li, S. Zhu, and M. Ogihara. Using discriminant analysis for multi-class classification: an experimental investigation. *Knowledge and information systems*, 10(4):453–472, 2006.

[35] K. Liu, Y.-Q. Cheng, J.-Y. Yang, and X. Liu. An efficient algorithm for foley–sammon optimal set of discriminant vectors by algebraic method. *International Journal of Pattern Recognition and Artificial Intelligence*, 6(05):817–829, 1992.

[36] J. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. Face recognition using lda-based algorithms. *IEEE Transactions on Neural Networks*, 14(1):195–200, 2003.

[37] J. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. Regularized discriminant analysis for the small sample size problem in face recognition. *Pattern Recognition Letters*, 24(16):3079–3087, 2003.

[38] J. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. Regularization studies of linear discriminant analysis in small sample size scenarios with application to face recognition. *Pattern Recognition Letters*, 26(2):181–191, 2005.

[39] B. Moghaddam, Y. Weiss, and S. Avidan. Generalized spectral bounds for sparse lda. In *Proceedings of the 23rd international conference on Machine learning*, pages 641–648. ACM, 2006.

[40] W. Müller, T. Nocke, and H. Schumann. Enhancing the visualization process with principal component analysis to support the exploration of trends. In *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation*, volume 60, pages 121–130. Australian Computer Society, Inc., 2006.

[41] S. Noushath, G. H. Kumar, and P. Shivakumara. (2d) 2 lda: An efficient approach for face recognition. *Pattern recognition*, 39(7):1396–1400, 2006.

[42] K. K. Paliwal and A. Sharma. Improved pseudoinverse linear discriminant analysis method for dimensionality reduction. *International Journal of Pattern Recognition and Artificial Intelligence*, 26(01):1250002, 2012.

[43] F. Pan, G. Song, X. Gan, and Q. Gu. Consistent feature selection and its application to face recognition. *Journal of Intelligent Information Systems*, 43(2):307–321, 2014.

[44] C. H. Park and H. Park. Fingerprint classification using fast fourier transform and nonlinear discriminant analysis. *Pattern Recognition*, 38(4):495–503, 2005.

[45] C. H. Park and H. Park. A comparison of generalized linear discriminant analysis algorithms. *Pattern Recognition*, 41(3):1083–1097, 2008.

[46] S. Rezzi, D. E. Axelson, K. Héberger, F. Reniero, C. Mariani, and C. Guillou. Classification of olive oils using high throughput flow 1 h nmr fingerprinting with principal component analysis, linear discriminant analysis and probabilistic neural networks. *Analytica Chimica Acta*, 552(1):13–24, 2005.

[47] Y. Saeys, I. Inza, and P. Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.

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

[49] B. Schölkopf, C. J. Burges, and A. J. Smola. *Advances in kernel methods: support vector learning*. MIT press, 1999.

[50] B. Schölkopf and K.-R. Mullert. Fisher discriminant analysis with kernels. In *Proceedings of the 1999 IEEE Signal Processing Society Workshop Neural Networks for Signal Processing IX, Madison, WI, USA*, pages 41–48, 1999.

[51] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[52] A. Sharma, S. Imoto, and S. Miyano. A between-class overlapping filter-based method for transcriptome data analysis. *Journal of bioinformatics and computational biology*, 10(05):1250010, 2012.

[53] A. Sharma, S. Imoto, and S. Miyano. A filter based feature selection algorithm using null space of covariance matrix for dna microarray gene expression data. *Current Bioinformatics*, 7(3):289–294, 2012.

[54] A. Sharma, S. Imoto, S. Miyano, and V. Sharma. Null space based feature selection method for gene expression data. *International Journal of Machine Learning and Cybernetics*, 3(4):269–276, 2012.

[55] A. Sharma and K. K. Paliwal. Cancer classification by gradient lda technique using microarray gene expression data. *Data & Knowledge Engineering*, 66(2):338–347, 2008.

[56] A. Sharma and K. K. Paliwal. A new perspective to null linear discriminant analysis method and its fast implementation using random matrix multiplication with scatter matrices. *Pattern Recognition*, 45(6):2205–2213, 2012.

[57] A. Sharma and K. K. Paliwal. Linear discriminant analysis for the small sample size problem: an overview. *International Journal of Machine Learning and Cybernetics*, pages 1–12, 2014.

[58] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.

[59] G. Strang. Introduction to linear algebra. *Wellesley-Cambridge Press, Massachusetts, Fourth Edition*, 2003.

[60] D. L. Swets and J. J. Weng. Using discriminant eigenfeatures for image retrieval. *IEEE Transactions on pattern analysis and machine intelligence*, 18(8):831–836, 1996.

[61] M. M. Tantawi, K. Revett, A. Salem, and M. F. Tolba. Fiducial feature reduction analysis for electrocardiogram (ecg) based biometric recognition. *Journal of Intelligent Information Systems*, 40(1):17–39, 2013.

[62] A. Tharwat. Principal component analysis-a tutorial. *International Journal of Applied Pattern Recognition*, 3(3):197–240, 2016.

[63] A. Tharwat, T. Gaber, Y. M. Awad, N. Dey, and A. E. Hassanien. Plants identification using feature fusion technique and bagging classifier. In *The 1st International Conference on Advanced Intelligent System and Informatics (AISI2015), November 28-30, 2015, Beni Suef, Egypt*, pages 461–471. Springer, 2016.

[64] A. Tharwat, T. Gaber, and A. E. Hassanien. One-dimensional vs. two-dimensional based features: Plant identification approach. *Journal of Applied Logic*, 2016.

[65] A. Tharwat, T. Gaber, A. E. Hassanien, H. A. Hassanien, and M. F. Tolba. Cattle identification using muzzle print images based on texture features approach. In *Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014*, pages 217–227. Springer, 2014.

[66] A. Tharwat, A. E. Hassanien, and B. E. Elnaghi. A ba-based algorithm for parameter optimization of support vector machine. *Pattern Recognition Letters*, 2016.

[67] A. Tharwat, A. Ibrahim, A. E. Hassanien, and G. Schaefer. Ear recognition using block-based principal component analysis and decision fusion. In *International Conference on Pattern Recognition and Machine Intelligence*, pages 246–254. Springer, 2015.

[68] A. Tharwat, H. Mahdi, A. El Hennawy, and A. E. Hassanien. Face sketch synthesis and recognition based on linear regression transformation and multi-classifier technique. In *The 1st International Conference on Advanced Intelligent System and Informatics (AISI2015), November 28-30, 2015, Beni Suef, Egypt*, pages 183–193. Springer, 2016.

[69] A. Tharwat, Y. S. Moemen, and A. E. Hassanien. Classification of toxicity effects of biotransformed hepatic drugs using whale optimized support vector machines. *Journal of Biomedical Informatics*, 2017.

[70] C. G. Thomas, R. A. Harshman, and R. S. Menon. Noise reduction in bold-based fmri using component analysis. *Neuroimage*, 17(3):1521–1537, 2002.

[71] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991.

[72] V. Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, Second Edition, 2013.

[73] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *The Journal of Machine Learning Research*, 11:451–490, 2010.

[74] P. Viszlay, M. Lojka, and J. Juhár. Class-dependent two-dimensional linear discriminant analysis using two-pass recognition strategy. In *Proceedings of the 22nd European Signal Processing Conference (EUSIPCO)*, pages 1796–1800. IEEE, 2014.

[75] X. Wang and X. Tang. Random sampling lda for face recognition. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages II–II. IEEE, 2004.

[76] M. Welling. Fisher linear discriminant analysis. *Department of Computer Science, University of Toronto*, 3, 2005.

[77] M. C. Wu, L. Zhang, Z. Wang, D. C. Christiani, and X. Lin. Sparse linear discriminant analysis for simultaneous testing for the significance of a gene set/pathway and gene selection. *Bioinformatics*, 25(9):1145–1151, 2009.

[78] J. Yang, D. Zhang, A. F. Frangi, and J.-y. Yang. Two-dimensional pca: a new approach to appearance-based face representation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):131–137, 2004.

[79] L. Yang, W. Gong, X. Gu, W. Li, and Y. Liang. Null space discriminant locality preserving projections for face recognition. *Neurocomputing*, 71(16):3644–3649, 2008.

[80] W. Yang and H. Wu. Regularized complete linear discriminant analysis. *Neurocomputing*, 137:185–191, 2014.

[81] J. Ye, R. Janardan, and Q. Li. Two-dimensional linear discriminant analysis. In *Proceedings of 17$^{th}$ Advances in Neural Information Processing Systems (NIPS)*, pages 1569–1576, 2004.

[82] J. Ye and T. Xiong. Computational and theoretical analysis of null space and orthogonal linear discriminant analysis. *The Journal of Machine Learning Research*, 7:1183–1204, 2006.

[83] H. Yu and J. Yang. A direct lda algorithm for high-dimensional data with application to face recognition. *Pattern recognition*, 34(10):2067–2070, 2001.

[84] L. Yuan and Z.-c. Mu. Ear recognition based on 2d images. In *Proceedings of the first IEEE International Conference on Biometrics: Theory, Applications, and Systems, 2007. BTAS 2007.*, pages 1–5. IEEE, 2007.

[85] X.-S. Zhuang and D.-Q. Dai. Inverse fisher discriminate criteria for small sample size problem and its application to face recognition. *Pattern Recognition*, 38(11):2192–2194, 2005.