Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 04-042

Comparitive Study of Linear and Nonlinear Feature Extraction Methods, A

Cheonghee Park, Haesun Park, and Panos Pardalos

November 17, 2004

A Comparative Study of Linear and Nonlinear Feature Extraction Methods

Cheong Hee Park[1] Haesun Park[2*] Panos Pardalos[3†]

Dept. of Computer Science and Engineering[1,2]

University of Minnesota

Minneapolis, MN 55455, USA

The National Science Foundation[2]

4201 Wilson Boulevard, Arlington, Virginia 22230, USA

Dept. of Industrial and Systems Engineering[3]

University of Florida

Gainesville, FL 32611, USA

[Extended version of Technical Report 03-048]

Abstract

Linear Discriminant Analysis (LDA) is a dimension reduction method which finds an optimal linear transformation that maximizes the between-class scatter and minimizes the within-class scatter. However, in undersampled problems where the number of samples is smaller than

*This work was supported in part by the National Science Foundation grants CCR-0204109 and ACI-0305543.

†The work was partially supported by NSF and NATO grants. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation (NSF). The work of Haesun Park has been performed while at the NSF and was partly supported by IR/D from the NSF.

the dimension of data space, it is difficult to apply the LDA due to the singularity of scatter matrices caused by high dimensionality. In order to make the LDA applicable, several generalizations of the LDA have been proposed. This paper presents theoretical and algorithmic relationships among several generalized LDA algorithms. Utilizing the relationships among them, computationally efficient approaches to these algorithms are proposed. We also present nonlinear extensions of these LDA algorithms. The original data space is mapped to a feature space by an implicit nonlinear mapping through kernel methods. A generalized eigenvalue problem is formulated in the transformed feature space and generalized LDA algorithms are applied to solve the problem. Performances and computational complexities of these linear and nonlinear discriminant analysis algorithms are compared theoretically and experimentally.
Keywords: Dimension reduction, Feature extraction, Generalized Linear Discriminant Analysis, Kernel methods, Nonlinear Discriminant Analysis, Undersampled problems.

1 Introduction

Linear Discriminant Analysis (LDA) seeks an optimal linear transformation by which the original data is transformed to a much lower dimensional space. The goal of LDA is to find a linear transformation that maximizes class separability in the reduced dimensional space. Hence the criteria for dimension reduction in LDA are formulated to maximize the between-class scatter and minimize the within-class scatter using scatter matrices. The scatters between classes, within each class, and among all data items are represented as the between-class scatter matrix (S_b), within-class scatter matrix (S_w) and total scatter matrix (S_t), respectively. Denoting a data set A as

$$A = [a_1, \cdots, a_n] = [A_1, A_2, \cdots, A_r] \in \mathbb{R}^{m \times n} \tag{1}$$

where a collection of data items in the class i is represented as a block matrix $A_i \in \mathbb{R}^{m \times n_i}$, each class i $(1 \leq i \leq r)$ has n_i elements and the total number of data is $n = \sum_{i=1}^{r} n_i$. Let N_i $(1 \leq i \leq r)$ be the index set of data items in the class i. Then the between-class scatter matrix S_b, within-class

scatter matrix S_w and total scatter matrix S_t are defined as

$$\begin{aligned} S_b &= \sum_{i=1}^{r} n_i (c_i - c)(c_i - c)^T \in \mathbb{R}^{m \times m}, \\ S_w &= \sum_{i=1}^{r} \sum_{j \in N_i} (a_j - c_i)(a_j - c_i)^T \in \mathbb{R}^{m \times m}, \\ S_t &= \sum_{j=1}^{n} (a_j - c)(a_j - c)^T \in \mathbb{R}^{m \times m} \end{aligned}$$

where $c_i = \frac{1}{n_i} \sum_{j \in N_i} a_j$ and $c = \frac{1}{n} \sum_{j=1}^{n} a_j$ are class centroids and the global centroid, respectively. The *traces* of scatter matrices can be used to measure the quality of clustered structure in the data as

$$\text{trace}(S_b) = \sum_{i=1}^{r} n_i \|c_i - c\|_2^2, \quad \text{trace}(S_w) = \sum_{i=1}^{r} \sum_{j \in N_i} \|a_j - c_i\|_2^2.$$

The distance between classes is quantified by trace(S_b), and trace(S_w) measures the scatters within classes.

The optimal dimension reducing transformation for LDA is the one that maximizes the between-class scatter and minimizes the within-class scatter in a reduced dimensional space. One of the common optimization criteria in LDA is to find a linear transformation $G^T \in \mathbb{R}^{l \times m}$ which maximizes

$$J(G) = \text{trace}((G^T S_w G)^{-1} (G^T S_b G)) \tag{2}$$

where $\tilde{S}_i = G^T S_i G$ for $i = b, w$ are scatter matrices in the space transformed by G^T. It is well known [1, 2] that when S_w is nonsingular, the transformation matrix G is obtained by the eigenvectors corresponding to the $r - 1$ largest eigenvalues of

$$S_w^{-1} S_b g = \lambda g. \tag{3}$$

However, for undersampled problems such as text classification and face recognition where the number of data items is smaller than the data dimension, scatter matrices become singular and their inverses are not defined. In order to overcome the problems caused by the singularity of the scatter matrices, several methods have been proposed [3, 2, 4, 5, 6], In this paper, we present theoretical relationships among several generalized LDA algorithms and compare computational complexities and performances.

While linear dimension reduction has been used in many application areas due to its simple concept and easiness in computation, it is difficult to capture a nonlinear relationship in the data by a linear function. We show that the generalized LDA algorithms can be extended to nonlinear discriminant analysis. Classification performances by nonlinear discriminant analysis are compared using data sets from UCI databases as well as undersampled problems.

The rest of this paper is organized as follows. In Section 2, a comparison of generalized LDA algorithms is presented. We study theoretical and algorithmic relationships among several generalized LDA algorithms and compare their computational complexities and performances. Computationally efficient approaches to them are also proposed which compute the exactly same solutions as those in [2, 4, 6] but save computational complexities greatly. In Section 3, nonlinear extensions of these generalized LDA algorithms are proposed. The original data space is mapped to a feature space by an implicit nonlinear mapping through kernel methods. A generalized eigenvalue problem is formulated in the feature space which enables us to apply all the available generalized LDA algorithms resulting in nonlinear dimension reduction methods. Extensive comparisons of these linear and nonlinear discriminant analysis algorithms are conducted. In Section 4, we conclude with discussions.

2 A Comparison of Generalized LDA Algorithms for Undersampled Problems

In this section, we present a theoretical comparison of several generalized LDA algorithms [3, 2, 4, 5, 6] and compare their computational complexities and performances. Utilizing the relationships discovered, we propose efficient approaches to the generalized LDA algorithms [2, 4, 6]. To make matrix notations consistent, subscripts are used as follows. The symmetric Eigenvalue Decomposition (EVD) of $S_i \in \mathbb{R}^{m \times m}$, $i = b, w, t$, is represented as

$$S_i = U_i \Sigma_i U_i^T = \begin{bmatrix} U_{i1} & U_{i2} \end{bmatrix} \begin{bmatrix} \Sigma_{i1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_{i1}^T \\ U_{i2}^T \end{bmatrix}$$

where $U_i \in \mathbb{R}^{m \times m}$ is orthogonal and $\Sigma_i \in \mathbb{R}^{m \times m}$ is a diagonal matrix with nonincreasing diagonal components. U_{i1} and U_{i2} are block matrices corresponding to nonzero eigenvalues and zero

eigenvalues respectively. We also note that the scatter matrices can be computed as

$$S_b = H_b H_b^T, \quad H_b = [\sqrt{n_1}(c_1 - c), \cdots, \sqrt{n_r}(c_r - c)] \in \mathbb{R}^{m \times r}, \tag{4}$$

$$S_w = H_w H_w^T, \quad H_w = [A_1 - c_1 e_1, \cdots, A_r - c_r e_r] \in \mathbb{R}^{m \times n}, \tag{5}$$

$$S_t = H_t H_t^T, \quad H_t = [a_1 - c, \cdots, a_n - c] \in \mathbb{R}^{m \times n}, \tag{6}$$

where $e_i = [1, \cdots, 1] \in \mathbb{R}^{1 \times n_i}$. For the methods discussed in this section, Table 5 summarizes the pseudocodes which were implemented in our experiments. The pseudocodes given in Table 5 were optimized in terms of computational complexities and memory requirements, especially for high dimensional data up to the data dimension of thousands.

2.1 Regularized LDA

In the regularized LDA (RLDA), when S_w is singular or ill-conditioned, a diagonal matrix αI with $\alpha > 0$ is added to S_w. Since S_w is symmetric positive semidefinite, $S_w + \alpha I$ is nonsingular with any $\alpha > 0$. Therefore we can apply the algorithm for the classical LDA to solve the eigenvalue problem

$$S_b g = \lambda (S_w + \alpha I) g. \tag{7}$$

The diagonalization of $S_w + \alpha I$ can be obtained from the EVD of $S_w \in \mathbb{R}^{m \times m}$ which can be computed by using the smaller matrix $H_w \in \mathbb{R}^{m \times n}$ as follows. Let the singular value decomposition (SVD) of H_w be

$$H_w = U_w D_w Y_w^T,$$

where $U_w \in \mathbb{R}^{m \times m}$ and $Y_w \in \mathbb{R}^{n \times n}$ are orthogonal and $D_w \in \mathbb{R}^{m \times n}$ is a diagonal matrix. Then the EVD of S_w is obtained as

$$S_w = H_w H_w^T = U_w \Sigma_w U_w^T \quad \text{where} \quad \Sigma_w = D_w D_w^T$$

and the EVD of $S_w + \alpha I$ as

$$S_w + \alpha I = U_w (\Sigma_w + \alpha I) U_w^T.$$

Now, from the EVD of $\tilde{S}_b \equiv (\Sigma_w + \alpha I)^{-1/2}\, U_w^T\, S_b\, U_w\, (\Sigma_w + \alpha I)^{-1/2}$,

$$\tilde{S}_b = \tilde{U}_b \tilde{\Sigma}_b \tilde{U}_b^T,$$

we obtain

$$V^T\, S_b\, V = \tilde{\Sigma}_b \quad \text{and} \quad V^T (S_w + \alpha I) V = I \tag{8}$$

where $V = U_w\, (\Sigma_w + \alpha I)^{-1/2} \tilde{U}_b$. Eqs. in (8) gives

$$S_b V = (S_w + \alpha I) V \tilde{\Sigma}_b,$$

which solves the problem (7), therefore the transformation matrix G is obtained by the first $r-1$ columns of $V = U_w(\Sigma_w + \alpha I)^{-1/2}\tilde{U}_b$. The smaller a value for the parameter α is, the greater the effect of the null space of S_w in computing the EVD of $\tilde{S}_b$ is.

Two-Class Problem

We now consider the simple case when the data set has two classes, since in that case the effect of the regularization parameter α to the solution is easy to illustrate. Two-class problem in LDA is known as Fisher Discriminant Analysis (FDA) [7]. In a two-class case, S_b can be expressed as

$$S_b = \frac{n_1 n_2}{n}(c_1 - c_2)(c_1 - c_2)^T, \tag{9}$$

and the eigenvalue problem (3) is simplified to

$$S_w^{-1}(c_1 - c_2)(c_1 - c_2)^T g = \lambda g \tag{10}$$

when S_w is nonsingular. The solution for (10) is a nonzero multiple of

$$g = S_w^{-1}(c_1 - c_2),$$

and the 1-dimensional representation of any data item $z \in \mathbb{R}^{m\times 1}$ by LDA is obtained as

$$g^T z = (c_1 - c_2)^T S_w^{-1} z = (c_1 - c_2)^T U_w \Sigma_w^{-1} U_w^T z.$$

Similarly, the regularized LDA by αI gives the solution

$$g^T z = (c_1 - c_2)^T U_w (\Sigma_w + \alpha I)^{-1} U_w^T z,$$

where the regularization parameter α affects the scales of the principal components of S_w.

In the regularized LDA, the parameter α is to be optimized experimentally since no theoretical procedure for choosing an optimal parameter is easily available. Recently, a generalization of LDA through simultaneous diagonalization of S_b and S_w using the generalized singular value decomposition (GSVD) has been developed [2]. This LDA/GSVD, summarized in the next section, does not require any parameter optimization.

2.2 LDA based on the Generalized Singular Value Decomposition

Howland et al. [2] applied the Generalized Singular Value Decomposition (GSVD) due to Paige and Saunders [8] to overcome the limitation of the classical LDA. When the GSVD is applied to two matrices Z_1 and Z_2 with the same number of columns, p , we obtain

$$U_1^T Z_1 X = [\underbrace{\Gamma_1}_{\gamma} \underbrace{0}_{p-\gamma}] \quad \text{and} \quad U_2^T Z_2 X = [\underbrace{\Gamma_2}_{\gamma} \underbrace{0}_{p-\gamma}] \quad \text{for} \quad \gamma = \text{rank}\left(\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix}\right)$$

where U_1 and U_2 are orthogonal and X is nonsingular, $\Gamma_1^T\Gamma_1 + \Gamma_2^T\Gamma_2 = I_\gamma$ and $\Gamma_1^T\Gamma_1$ and $\Gamma_2^T\Gamma_2$ are diagonal matrices with nonincreasing and nondecreasing diagonal components respectively.

The method in [2] utilized the representations of the scatter matrices

$$S_b = H_b H_b^T, \quad S_w = H_w H_w^T.$$

Suppose the GSVD is applied to the matrix pair $(H_b^T,\ H_w^T)$ and we obtain

$$U_b^T H_b^T X = [\Gamma_b \quad 0] \quad \text{and} \quad U_w^T H_w^T X = [\Gamma_w \quad 0]. \tag{11}$$

From (11) and $\Gamma_b^T\Gamma_b + \Gamma_w^T\Gamma_w = I_s$ for $s = \text{rank}\left(\begin{bmatrix} H_b^T \\ H_w^T \end{bmatrix}\right)$,

$$X^T S_b X = \begin{bmatrix} \Gamma_b^T\Gamma_b & & \\ & & \\ & & 0_{m-s} \end{bmatrix} \equiv \begin{bmatrix} I_\mu & & & \\ & D_\tau & & \\ & & 0_{s-\mu-\tau} & \\ & & & 0_{m-s} \end{bmatrix} \text{ and} \tag{12}$$

$$X^T S_w X = \begin{bmatrix} \Gamma_w^T\Gamma_w & & \\ & & \\ & & 0_{m-s} \end{bmatrix} \equiv \begin{bmatrix} 0_\mu & & & \\ & E_\tau & & \\ & & I_{s-\mu-\tau} & \\ & & & 0_{m-s} \end{bmatrix}, \tag{13}$$

where the subscripts in I and 0 denote the size of square identity and zero matrices. Denoting the diagonal elements in $\Gamma_b^T \Gamma_b$ as η_i's and the diagonal elements in $\Gamma_w^T \Gamma_w$ as ζ_i's, we have

$$\zeta_i S_b x_i = \eta_i S_w x_i \quad i = 1, \cdots, m, \tag{14}$$

where x_i is the column vectors of X. Note that x_i, $i = s+1, \cdots, m$, belong to null$(S_b) \cap$ null(S_w). Hence η_i and ζ_i for $i = s+1, \cdots, m$ in Eq. (14) can be any arbitrary numbers.

By partitioning X in (12 - 13) as

$$X = [\underbrace{X_1}_{\mu} \underbrace{X_2}_{\tau} \underbrace{X_3}_{s-\mu-\tau} \underbrace{X_4}_{m-s}], \tag{15}$$

the generalized eigenvalues and eigenvectors obtained by the GSVD can be classified as shown in Table 1. For the last $m - s$ vectors x belonging to null$(S_w) \cap$ null(S_b),

$$\begin{aligned} 0 &= x^T S_b x = (x^T H_b)(H_b^T x) = \|x^T H_b\|^2 = \sum_{i=1}^{r} n_i |x^T c_i - x^T c|^2 \quad \text{and} \\ 0 &= x^T S_w x = \sum_{j=1}^{n} |x^T a_j - x^T c_i|^2 \quad \text{where } a_j \text{ belongs to a class } i. \end{aligned}$$

Hence

$$\begin{cases} x^T c_i = x^T c \quad \text{for} \quad i = 1, \cdots, r \\ x^T a_j = x^T c_i \quad \text{for all} \;\; a_j \text{ in a class } i, \end{cases} \tag{16}$$

therefore

$$X_4^T z = X_4^T c \tag{17}$$

for any given data item $z = a_i$. This implies that the vectors x_i, $i = s+1, \cdots, m$, belonging to null$(S_b) \cap$ null(S_w) do not convey discriminative information among the classes, even though the corresponding eigenvalues are not necessarily zeros. Since rank$(S_b) \leq r - 1$, from Eqs. (12-13)

$$x_i^T S_b x_i = 0 \quad \text{and} \quad x_i^T S_w x_i = 1 \quad \text{for } r \leq i \leq s,$$

and the between-class scatter becomes zero by the projection onto the vector x_i. Hence $r - 1$ leftmost columns of X gives an optimal transformation G_h^T for LDA. This method is called LDA/GSVD.

	η_i	ζ_i	$\lambda_i = \frac{\eta_i}{\zeta_i}$	x_i belongs to
$1 \le i \le \mu$	1	0	∞	null$(S_w) \cap$ null$(S_b)^c$
$\mu + 1 \le i \le \mu + \tau$	$1 > \eta_i > 0$	$0 < \zeta_i < 1$	$\infty > \lambda_i > 0$	null$(S_w)^c \cap$ null$(S_b)^c$
$\mu + \tau + 1 \le i \le s$	0	1	0	null$(S_w)^c \cap$ null(S_b)
$s + 1 \le i \le m$	any value	any value	any value	null$(S_w) \cap$ null(S_b)

Table 1: Generalized eigenvalues λ_i's and eigenvectors x_i's from the GSVD. The superscript c denotes the complement.

A New Approach for LDA/GSVD

The algorithm to compute the GSVD for the pair (H_b^T, H_w^T) was presented in [2] as follows.

1. Compute the SVD of $Z = \begin{bmatrix} H_b^T \\ H_w^T \end{bmatrix} \in \mathbb{R}^{(r+n)\times m}$: $Z = P \begin{bmatrix} \Lambda & 0 \\ 0 & 0 \end{bmatrix} U^T$ where $s = \text{rank}(Z)$ and $P \in \mathbb{R}^{(r+n)\times(r+n)}$ and $U \in \mathbb{R}^{m\times m}$ are orthogonal and the diagonal components of $\Lambda \in \mathbb{R}^{s\times s}$ is nonincreasing.

2. Compute V from the SVD of $P(1:r, 1:s)$, which is $P(1:r, 1:s) = W\Gamma V^T$.

3. Compute the first $r-1$ columns of $X = U \begin{bmatrix} \Lambda^{-1}V & 0 \\ 0 & I \end{bmatrix}$, and assign them to the transformation matrix G_h.

Now we show that this algorithm can be computed rather simply, producing an efficient and intuitive approach for LDA/GSVD. Since $\Gamma_b^T\Gamma_b + \Gamma_w^T\Gamma_w = I_s$, from (12-13), we have

$$X^T S_t X = X^T S_b X + X^T S_w X = \begin{bmatrix} I_s & 0 \\ 0 & 0 \end{bmatrix} \tag{18}$$

where $s = \text{rank}(Z)$. Eq. (18) implies $s = \text{rank}(S_t)$ and from the step 3 in the LDA/GSVD algorithm

$$S_t = X^{-T} \begin{bmatrix} I_s & 0 \\ 0 & 0 \end{bmatrix} X^{-1} = U \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix} U^T, \quad \Sigma_1 = \Lambda^T\Lambda \tag{19}$$

which results in the EVD of S_t. Partitioning U as

$$U = [\underbrace{U_1}_{s} \underbrace{U_2}_{m-s}],$$

Algorithm 1 An efficient algorithm for LDA/GSVD

1. Compute the EVD of S_t : $S_t = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_1^T \\ U_2^T \end{bmatrix}$.
2. Compute V from the EVD of $\tilde{S}_b \equiv \Sigma_1^{-1/2} U_1^T S_b U_1 \Sigma_1^{-1/2}$: $\tilde{S}_b = V \Gamma_b^T \Gamma_b V^T$.
3. Assign the first $r-1$ columns of $U_1 \Sigma_1^{-1/2} V$ to G_h.

we have

$$X = U \begin{bmatrix} \Lambda^{-1} V & 0 \\ 0 & I \end{bmatrix} = \begin{bmatrix} U_1 \Sigma_1^{-1/2} V & U_2 \end{bmatrix}. \tag{20}$$

By substituting X in (12) with Eq. (20),

$$\Sigma_1^{-1/2} U_1^T S_b U_1 \Sigma_1^{-1/2} = V \Gamma_b^T \Gamma_b V^T. \tag{21}$$

Note that the optimal transformation matrix G_h by LDA/GSVD is obtained by the leftmost $r-1$ columns of X, which are the leftmost $r-1$ columns of $U_1 \Sigma_1^{-1/2} V$. Eqs. (19) and (21) show that U_1 and Σ_1 can be computed from the EVD of S_t and V from the EVD of $\Sigma_1^{-1/2} U_1^T S_b U_1 \Sigma_1^{-1/2}$. This new approach for LDA/GSVD is summarized in Algorithm 1.

In Algorithm 1, the matrices U_1 and Σ_1 in the EVD of $S_t \in \mathbb{R}^{m \times m}$ can be obtained by the EVD of $H_t^T H_t \in \mathbb{R}^{n \times n}$ instead of $H_t H_t^T \in \mathbb{R}^{m \times m}$ [1] by which computational complexity can be reduced from $O(m^3)$ to $O(n^3)$. Especially when m is much bigger than n, computational savings become great. Let the EVD of $H_t^T H_t$ be

$$H_t^T H_t = \begin{bmatrix} \underbrace{J_1}_{s} & \underbrace{J_2}_{n-s} \end{bmatrix} \begin{bmatrix} D_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} J_1^T \\ J_2^T \end{bmatrix}, \tag{22}$$

where $s = \text{rank}(H_t) = \text{rank}(S_t)$. From (22)

$$S_t (H_t J_1) = H_t (H_t^T H_t) J_1 = (H_t J_1) D_1,$$

and therefore the columns in $H_t J_1$ are eigenvectors of S_t corresponding to nonzero eigenvalues in the diagonal of D_1. Since

$$(H_t J_1)^T (H_t J_1) = D_1,$$

we obtain the orthonormal eigenvectors and corresponding nonzero eigenvalues of S_t by $H_t J_1 D_1^{-1/2}$ and D_1, which are U_1 and Σ_1 respectively. In this new approach, we just need to compute the EVD of a much smaller $n \times n$ matrix $H_t^T H_t$ instead of $m \times m$ matrix $S_t = H_t H_t^T$ when $m >> n$. However, in the regularized LDA or the method by Chen et al. which is presented next, we can not resort to this approach. The regularized LDA needs the entire m eigenvectors of S_w and the method based on the projection to null(S_w) needs to compute a basis of null(S_w) which are eigenvectors corresponding to zero eigenvalues.

Two-Class Problem

Now we consider two-class problem in LDA/GSVD. With the representation of S_b in (9), we have

$$\begin{aligned} \Sigma_1^{-1/2} U_1^T S_b U_1 \Sigma_1^{-1/2} &= \Sigma_1^{-1/2} U_1^T \rho (c_1 - c_2)(c_1 - c_2)^T U_1 \Sigma_1^{-1/2} \\ &= \left(\frac{w}{\|w\|_2} \right) \rho \|w\|_2^2 \left(\frac{w}{\|w\|_2} \right)^T, \end{aligned}$$

where $\rho = n_1 n_2 / n$ and $w = \Sigma_1^{-1/2} U_1^T (c_1 - c_2)$. Hence the transformation matrix $g \in \mathbb{R}^{m \times 1}$ is given by

$$g = \nu U_1 \Sigma_1^{-1/2} w = \nu U_1 \Sigma_1^{-1} U_1^T (c_1 - c_2)$$

for some scalar ν, and the dimension reduced representation of any data item z is given by

$$g^T z = \nu (c_1 - c_2)^T U_1 \Sigma_1^{-1} U_1^T z = \nu (c_1 - c_2)^T S_t^+ z,$$

where S_t^+ denotes the pseudoinverse of S_t. When S_w is nonsingular, by applying the Sherman-Morrison formula [9] to $S_t = S_w + S_b$, we have

$$S_t^{-1} = (S_w + \rho (c_1 - c_2)(c_1 - c_2)^T)^{-1} = S_w^{-1} - \frac{S_w^{-1} \rho (c_1 - c_2)(c_1 - c_2)^T S_w^{-1}}{1 + \rho (c_1 - c_2)^T S_w^{-1} (c_1 - c_2)}$$

and

$$g^T z = \nu (c_1 - c_2)^T S_t^{-1} z = \nu_1 (c_1 - c_2)^T S_w^{-1} z \tag{23}$$

for a scalar $\nu_1 = \nu / (1 + \rho (c_1 - c_2)^T S_w^{-1} (c_1 - c_2))$. Eq. (23) shows that LDA/GSVD is equal to the classical LDA when S_w is nonsingular.

Both regularization method and LDA/GSVD achieve the solution of LDA through the simultaneous diagonalization of scatter matrices, while the within-class scatter matrix handled in regularized LDA is modified to be nonsingular. In face recognition, in the efforts to overcome the singularity of scatter matrices caused by high dimensionality, some methods have been proposed [4, 5]. The basic principle of the algorithms proposed in [4, 5] is that the transformation using a basis of either range(S_b) or null(S_w) is performed in the first stage and then in the transformed space the second projective directions are searched. These methods are summarized in the next two sections where we also present their algebraic relationships.

2.3 A Method based on the Projection onto null(S_w)

Chen et al. [4] proposed a generalized method of LDA which solves undersampled problems and applied it for face recognition. The method projects the original space onto the null space of S_w using an orthonormal basis of null(S_w), and then in the projected space, a transformation that maximizes the between-class scatter is computed.

Consider the SVD of $S_w \in \mathbb{R}^{m\times m}$,

$$S_w = U_w \Sigma_w U_w^T.$$

Partitioning U_w as $U_w = [\underbrace{U_{w1}}_{s_1}\ \underbrace{U_{w2}}_{m-s_1}]$ where $s_1 = \text{rank}(S_w)$,

$$\text{null}(S_w) = \text{span}(U_{w2}). \tag{24}$$

First, the transformation by $U_{w2}U_{w2}^T$ projects the original data to null(S_w). Then, the eigenvectors corresponding to the largest eigenvalues of the between-class scatter matrix $\tilde{S}_b$ in the projected space are found. Let the EVD of $\tilde{S}_b \equiv U_{w2}U_{w2}^T S_b U_{w2}U_{w2}^T$ be

$$\tilde{S}_b = \tilde{U}_b\tilde{\Sigma}_b\tilde{U}_b^T = [\underbrace{\tilde{U}_{b1}}_{s_2}\ \underbrace{\tilde{U}_{b2}}_{m-s_2}] \begin{bmatrix} \tilde{\Sigma}_{b1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{U}_{b1}^T \\ \tilde{U}_{b2}^T \end{bmatrix}, \tag{25}$$

where $\tilde{U}_b^T\tilde{U}_b = I$, $s_2 = \text{rank}(\tilde{S}_b)$ and $\tilde{\Sigma}_{b1} \in \mathbb{R}^{s_2\times s_2}$. Then, the transformation matrix G_e is obtained by

$$G_e = U_{w2}U_{w2}^T\tilde{U}_{b1}. \tag{26}$$

Let us call this method *To-$N(S_w)$* as an abbreviation.

Two-Class Problem

In two-class problem, S_b is expressed as in (9) and

$$\tilde{S}_b = U_{w2}U_{w2}^T\rho(c_1-c_2)(c_1-c_2)^T U_{w2}U_{w2}^T = \begin{array}{cc} \frac{w}{\|w\|_2} & \rho\|w\|_2^2 \end{array} \begin{array}{c} \frac{w}{\|w\|_2} \end{array}^T$$

where $\rho = n_1 n_2/n$ and $w = U_{w2}U_{w2}^T(c_1-c_2) \in \mathbb{R}^{m\times 1}$. Hence the transformation matrix $g \in \mathbb{R}^{m\times 1}$ is obtained by

$$g = U_{w2}U_{w2}^T\frac{w}{\|w\|_2} = \nu U_{w2}U_{w2}^T(c_1-c_2)$$

with $\nu = 1/\|w\|_2$. For any data item $z \in \mathbb{R}^{m\times 1}$, the dimension reduced representation is given by

$$g^T z = \nu(c_1-c_2)^T U_{w2}U_{w2}^T z.$$

Relationship with LDA/GSVD

From (25), we have

$$\begin{bmatrix} \tilde{U}_{b1}^T \\ \tilde{U}_{b2}^T \end{bmatrix} U_{w2}\, U_{w2}^T\, S_b\, U_{w2}\, U_{w2}^T\, [\tilde{U}_{b1}\, \tilde{U}_{b2}] = \begin{bmatrix} \tilde{\Sigma}_{b1} & 0 \\ 0 & 0 \end{bmatrix}, \tag{27}$$

$$\begin{bmatrix} \tilde{U}_{b1}^T \\ \tilde{U}_{b2}^T \end{bmatrix} U_{w2}\, U_{w2}^T\, S_w\, U_{w2}\, U_{w2}^T\, [\tilde{U}_{b1}\, \tilde{U}_{b2}] = 0. \tag{28}$$

The second equation holds due to (24). Eqs. in (27-28) imply that the column vectors of G_e given in (26) belong to null$(S_w) \cap$null$(S_b)^c$ and they are discriminative vectors, since the transformation by these vectors minimizes the within-class scatter to zero and increases the between-class scatter. The top row of Table 1 shows that the LDA/GSVD solution also includes the vectors from null$(S_w) \cap$ null$(S_b)^c$. Based on this observation, this method *To-*$N(S_w)$ can be compared with LDA/GSVD. By denoting X in LDA/GSVD as

$$X = [\underbrace{X_1}_{\mu}\ \underbrace{X_2}_{\tau}\ \underbrace{X_3}_{s-\mu-\tau}\ \underbrace{X_4}_{m-s}], \tag{29}$$

we find a relationship between X_1 and $G_e = U_{w2}U_{w2}^T\tilde{U}_{b1}$.

Eq. (13) implies that $[X_1\ \ X_4]$ is a basis of null(S_w). Hence any vector in null(S_w) can be represented as a linear combination of column vectors in $[X_1\ \ X_4]$. The following Theorem shows the condition for any vector in null(S_w) to belong to null$(S_w) \cap$ null$(S_b)^c$.

THEOREM 1 *Any vector x belongs to null$(S_w) \cap$ null$(S_b)^c$ if and only if x is represented as $X_1 h + X_4 k$ where $h \neq 0 \in \mathbb{R}^{\mu \times 1}$ and $k \in \mathbb{R}^{(m-s)\times 1}$.*

Proof. Let $x \in \text{null}(S_w) \cap \text{null}(S_b)^c$. Since $[X_1 \ X_4]$ is a basis of null(S_w), $x = X_1 h + X_4 k$ for some $h \in \mathbb{R}^{\mu \times 1}$ and $k \in \mathbb{R}^{(m-s)\times 1}$. Suppose $h = 0$. Then $x = X_4 k \in \text{null}(S_w) \cap \text{null}(S_b)$, which contradicts to $x \in \text{null}(S_w) \cap \text{null}(S_b)^c$. Hence $h \neq 0$.

Now let us prove that if $h \neq 0$ then $x = X_1 h + X_4 k$ belongs to null$(S_w) \cap$ null$(S_b)^c$. Since $x = X_1 h + X_4 k \in \text{null}(S_w)$, it is enough to show $x \notin \text{null}(S_b)$. From (12),

$$x^T S_b x = (X_1 h)^T S_b (X_1 h) = h^T (X_1^T S_b X_1) h = h^T I_\mu h = \|h\|_2^2 \neq 0. \quad \square$$

By Theorem 1,

$$U_{w2} U_{w2}^T \tilde{U}_{b1} = X_1 H + X_4 K$$

for some matrices $H \in \mathbb{R}^{\mu \times s_2}$ and $K \in \mathbb{R}^{(m-s)\times s_2}$ with $s_2 = \text{rank}(\tilde{S}_b)$, where each column of H is nonzero. Hence for any data item $z \in \mathbb{R}^{m \times 1}$, the reduced dimensional representation by $G_e = U_{w2} U_{w2}^T \tilde{U}_{b1}$ is given as

$$G_e^T z = H^T X_1^T z + K^T X_4^T z. \tag{30}$$

As explained in (16) of Section 2.2, since all data items are transformed to one point by x^T for $x \in \text{null}(S_w) \cap \text{null}(S_b)$, the second part $K^T X_4^T z$ in (30) corresponds to the translation which does not affect the classification performance.

While the transformation matrix $G_e = U_{w2} U_{w2}^T \tilde{U}_{b1}$ by the method *To-$N(S_w)$* is related to X_1 of LDA/GSVD as in (30), the main difference between the two methods is due to the eigenvectors in null$(S_w)^c \cap$ null$(S_b)^c$, which correspond to the second row in Table 1. The projection to null(S_w) by $U_{w2} U_{w2}^T$ excludes vectors in null$(S_w)^c$, and therefore null$(S_w)^c \cap$ null$(S_b)^c$. When

$$\text{rank}(\tilde{S}_b) < \text{rank}(S_b) \leq r - 1$$

where r is the number of classes, the reduced dimension by $G_e = U_{w2} U_{w2}^T \tilde{U}_{b1}$ is rank$(\tilde{S}_b)$, therefore less than $r - 1$, while LDA/GSVD includes $r - 1$ vectors from both null$(S_w) \cap$ null$(S_b)^c$ and null$(S_w)^c \cap$ null$(S_b)^c$. In order to demonstrate this case, we conducted an experiment using data in text classification, of which characteristics will be discussed in detail in the section for experiments.

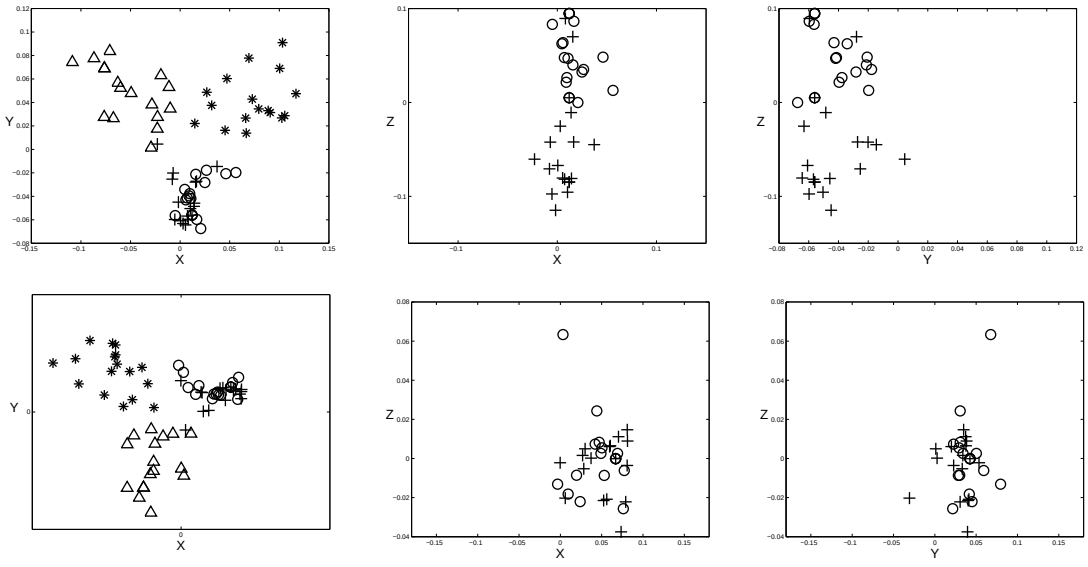

Figure 1: The visualization of the data in the reduced dimensional spaces by LDA/GSVD (figures in the first row) and the method *To-*$N(S_w)$ (figures in the second row).

The data was collected from Reuters-21578 database and contains 4 classes. Each class has 80 samples and the data dimension is 2412. After splitting the dataset randomly to training data and test data with a ratio of 4:1, the linear transformations by LDA/GSVD and the method *To-*$N(S_w)$ were computed by using training data. While the rank of S_b was 3, the rank of $\tilde{S}_b$ was 2 in this dataset. Hence the reduced dimension by the method *To-*$N(S_w)$ due to Chen et al. was 2. On the other hand, LDA/GSVD produced two eigenvectors from null$(S_w) \cap$ null$(S_b)^c$ and one eigenvector from null$(S_w)^c \cap$ null$(S_b)^c$, resulting in the reduced dimension 3. Figure 1 illustrates the reduced dimensional spaces by both methods. The top three figures were generated by LDA/GSVD. For the visualization, the data reduced to 3-dimensional space by LDA/GSVD was projected to 2-dimensional spaces, x-y, x-z and y-z spaces, respectively. In x-y space, two classes ($\triangle$ and *) are well separated, while two other classes (O and +) are mixed together. However, as shown in the second and third figures, two classes mixed in x-y space are separated in x-z and y-z spaces along z axis. This shows the third eigenvector from null$(S_w)^c \cap$ null$(S_b)^c$ improves the separation of classes. The bottom three figures were generated by the method based on the projection to null(S_w). Since rank$(\tilde{S}_b)$=2, the reduced dimension by that method was 2 and the first figure illustrates the reduced dimensional space. The second and third figures show that adding one more column vector from $U_{w2}U_{w2}^T\tilde{U}_{b2}$ and increasing the reduced dimension to 3 does not improve the

separation of classes mixed in x-y space, since the one extra dimension comes from null$(S_w) \cap$ null(S_b). On the other hand, when

$$\text{rank}(\tilde{S}_b) = \text{rank}(S_b) = r - 1,$$

both LDA/GSVD and the method *To-$N(S_w)$* obtain transformation matrices G_h and G_e from null$(S_w) \cap$ null$(S_b)^c$. Then the difference between two methods comes from the diagonal components of I_{r-1} and $\tilde{\Sigma}_{b1}$ in

$$G_h^T S_b G_h = I_{r-1} \quad \text{and} \quad G_e^T S_b G_e = \tilde{\Sigma}_{b1}$$

where $\tilde{\Sigma}_{b1}$ has nonincreasing diagonal components. As shown in the experimental results of Section 2.7, the effects of different scaling in the diagonal components may depend on the characteristics of data.

2.4 A Method based on the Transformation by a Basis of range(S_b)

In this section, we review another two-step approach by Yu and Yang [5] proposed to handle undersampled problems, and illustrate its relationship to other methods. Contrary to the method discussed in Section 2.3, the method presented in this section first transforms the original space by using a basis of range(S_b), and then in the transformed space the minimization of within-class scatter is pursued.

Consider the EVD of S_b,

$$S_b = U_b \Sigma_b U_b^T = [\underbrace{U_{b1}}_{s_1} \underbrace{U_{b2}}_{m-s_1}] \begin{bmatrix} \Sigma_{b1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_{b1}^T \\ U_{b2}^T \end{bmatrix},$$

where U_b is orthogonal, rank$(S_b) = s_1$ and Σ_{b1} is a diagonal matrix with nonincreasing positive diagonal components. Then

$$\text{range}(S_b) = \text{span}(U_{b1}).$$

In the method by Yu and Yang, the original data is first transformed to an s_1-dimensional space by $V_y = U_{b1}\, \Sigma_{b1}^{-1/2}$. Then the between-class scatter matrix $\tilde{S}_b$ in the transformed space becomes

$$\tilde{S}_b \equiv V_y^T S_b V_y = I_{s_1}.$$

Now consider the EVD of $\tilde{S}_w \equiv V_y^T S_w V_y$,

$$\tilde{S}_w = \tilde{U}_w \tilde{\Sigma}_w \tilde{U}_w^T, \tag{31}$$

where $\tilde{U}_w \in \mathbb{R}^{s_1 \times s_1}$ is orthogonal and $\tilde{\Sigma}_w \in \mathbb{R}^{s_1 \times s_1}$ is a diagonal matrix. Then

$$\tilde{U}_w^T V_y^T S_b V_y \tilde{U}_w = I_{s_1} \quad \text{and} \quad \tilde{U}_w^T V_y^T S_w V_y \tilde{U}_w = \tilde{\Sigma}_w. \tag{32}$$

In most applications, rank(S_w) is greater than rank(S_b), and $\tilde{\Sigma}_w$ is nonsingular since

$$\text{rank}(\tilde{U}_w^T V_y^T S_w V_y \tilde{U}_w) = \text{rank}(S_w) \geq \text{rank}(S_b) = \text{rank}(\tilde{U}_w^T V_y^T S_b V_y \tilde{U}_w) = s_1.$$

Scaling (32) by $\tilde{\Sigma}_w^{-1/2}$, we have

$$(\tilde{\Sigma}_w^{-1/2} \tilde{U}_w^T V_y^T) S_b (V_y \tilde{U}_w \tilde{\Sigma}_w^{-1/2}) = \tilde{\Sigma}_w^{-1}, \quad (\tilde{\Sigma}_w^{-1/2} \tilde{U}_w^T V_y^T) S_w (V_y \tilde{U}_w \tilde{\Sigma}_w^{-1/2}) = I_{s_1}. \tag{33}$$

The authors in [5] proposed the transformation matrix

$$G_y = V_y \tilde{U}_w \tilde{\Sigma}_w^{-1/2}.$$

Eqs. in (33) imply that each column of G_y belongs to null$(S_w)^c \cap$ null$(S_b)^c$. We call this method *To-$R(S_b)$* for short.

Two-Class Problem

In a two-class problem, since

$$S_b = \rho(c_1 - c_2)(c_1 - c_2)^T = \begin{bmatrix} \dfrac{c_1 - c_2}{\|c_1 - c_2\|_2} & \rho\|c_1 - c_2\|_2^2 & \dfrac{c_1 - c_2}{\|c_1 - c_2\|_2} \end{bmatrix}^T$$

where $\rho = n_1 n_2 / n$, a data item is transformed to the 1-dimensional space by

$$g^T = \begin{bmatrix} \dfrac{c_1 - c_2}{\sqrt{\rho}\|c_1 - c_2\|_2^2} \end{bmatrix}^T.$$

The dimension reduced representation of any data item z is given by

$$g^T z = \nu(c_1 - c_2)^T z$$

for some scalar ν. Note that no minimization of within-class scatter in the transformed space is possible.

	Transformation matrix		
Face data	$G_y = V_y$	$G_y = V_y\tilde{U}_w$	$G_y = V_y\tilde{U}_w\tilde{\Sigma}_w^{-1/2}$
AT&T	94.3	94.3	99.0
Yale	80.6	80.6	89.7

Table 2: The prediction accuracies(%).

The optimization criterion in (2) is invariant under any nonsingular linear transformation, i.e. for any nonsingular matrix F whose order is the same as that of the column dimension of G,

$$J_2(G) = J_2(GF). \tag{34}$$

Hence in the transformation matrix

$$G_y = V_y\tilde{U}_w\tilde{\Sigma}_w^{-1/2}$$

obtained by the method *To-*$R(S_b)$, $\tilde{\Sigma}_w^{-1/2}$ or $\tilde{U}_w\tilde{\Sigma}_w^{-1/2}$ do not influence the value of the optimization criterion, since

$$J_2(V_y) = J_2(V_y\tilde{U}_w) = J_2(V_y\tilde{U}_w\tilde{\Sigma}_w^{-1/2}).$$

Therefore, none of the components involved in the second step (those in (31-33)) improves the optimization criterion. However, the following experimental results show that the scaling by $\tilde{\Sigma}_w^{-1/2}$ can make dramatic effects on the classification performances. Postponing the detailed explanation on the datasets and experimental setting until Section 2.7, experimental results on the face recognition datasets are shown in Table 2. After dimension reduction, 1-NN classifier was used in the reduced dimensional space.

2.5 A Method of PCA plus Transformations to range(S_w) and null(S_w)

As shown in the analysis of the methods compared, they search for discriminative vectors in null$(S_w)\cap$ null$(S_b)^c$ and null$(S_w)^c\cap$ null$(S_b)^c$. The method *To-*$N(S_w)$ by Chen et al. finds solution vectors in null$(S_w)\cap$ null$(S_b)^c$ and *To-*$R(S_b)$ by Yu et al. restricts the search space to null$(S_w)^c\cap$ null$(S_b)^c$. LDA/GSVD by Howland et al. finds solution from both spaces, however the number of

possible discriminative vectors can not be greater than rank(S_b), possibly resulting in solution vectors only from null(S_w)$\cap$ null$(S_b)^c$ in the case of high dimensional data. Recently Yang et al. [6] have proposed a method to obtain solution vectors in both spaces, which we will call *To-$NR(S_w)$* .

In the method by Yang et al., first, the transformation by the orthonormal basis of range(S_t), as in PCA, is performed. Let the SVD of S_t be

$$S_t = U_t \Sigma_t U_t^T = [\underbrace{U_{t1}}_{s} \underbrace{U_{t2}}_{m-s}] \begin{bmatrix} \Sigma_{t1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_{t1}^T \\ U_{t2}^T \end{bmatrix}$$

where $s = \text{rank}(S_t)$. In the transformed space by U_{t1}, let the within-scatter matrix be $\tilde{S}_w = U_{t1}^T S_w U_{t1}$. Then the basis of null($\tilde{S}_w$) and range($\tilde{S}_w$) can be found by the EVD of $\tilde{S}_w$ as

$$\tilde{S}_w = \tilde{U}_w \tilde{\Sigma}_w \tilde{U}_w^T = [\tilde{U}_{w1}\, \tilde{U}_{w2}] \begin{bmatrix} \tilde{\Sigma}_{w1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{U}_{w1}^T \\ \tilde{U}_{w2}^T \end{bmatrix}. \tag{35}$$

In the transformed space by the basis $\tilde{U}_{w2}$ of null($\tilde{S}_w$), let Y be the matrix whose columns are the eigenvectors corresponding to nonzero eigenvalues of

$$\bar{S}_b \equiv \tilde{U}_{w2}^T U_{t1}^T S_b U_{t1} \tilde{U}_{w2}. \tag{36}$$

On the other hand, in the transformed space by the basis $\tilde{U}_{w1}$ of range($\tilde{S}_w$), let Z be the matrix whose columns are the eigenvectors[1] with the k largest nonzero eigenvalues[2] of $\hat{S}_t^{-1}\hat{S}_b$ where $\hat{S}_b \equiv \tilde{U}_{w1}^T U_{t1}^T S_b U_{t1} \tilde{U}_{w1}$ and $\hat{S}_t \equiv \tilde{U}_{w1}^T U_{t1}^T S_t U_{t1} \tilde{U}_{w1}$. Then the transformation matrix by the method *To-$NR(S_w)$*is constructed as

$$G_d = [U_{t1}\tilde{U}_{w2}Y \quad U_{t1}\tilde{U}_{w1}Z]. \tag{37}$$

It is obvious that the first part $U_{t1}\tilde{U}_{w2}Y$ in G_d comes from null(S_w)$\cap$ null$(S_b)^c$.

Now we show that eigenvectors of $\hat{S}_t^{-1}\hat{S}_b$ can be obtained easily as follows. By (35),

$$\hat{S}_w \equiv \tilde{U}_{w1}^T \tilde{S}_w \tilde{U}_{w1} = \tilde{U}_{w1}^T U_{t1}^T S_w U_{t1} \tilde{U}_{w1} = \tilde{\Sigma}_{w1}. \tag{38}$$

[1]In [6], it was claimed that the orthonormal eigenvectors of $\hat{S}_t^{-1}\hat{S}_b$ should be used. However, $\hat{S}_t^{-1}\hat{S}_b$ may not be symmetric therefore it is not guaranteed that there exist orthonormal eigenvectors of $\hat{S}_t^{-1}\hat{S}_b$.

[2]Here k should be determined experimentally. Instead, in our experiments for the comparison with other methods, the rank of $\hat{S}_b$ was used for the value l.

Let the EVD of $\tilde{\Sigma}_{w1}^{-1/2}\hat{S}_b\tilde{\Sigma}_{w1}^{-1/2}$ be

$$\tilde{\Sigma}_{w1}^{-1/2}\hat{S}_b\tilde{\Sigma}_{w1}^{-1/2} = \hat{U}_b\hat{\Sigma}_b\hat{U}_b^T = [\hat{U}_{b1}\,\hat{U}_{b2}]\begin{bmatrix}\hat{\Sigma}_{b1} & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}\hat{U}_{b1}^T\\ \hat{U}_{b2}^T\end{bmatrix}. \tag{39}$$

Then by Eqs. (38) and (39)

$$\hat{S}_t = \hat{S}_w + \hat{S}_b = \tilde{\Sigma}_{w1} + \tilde{\Sigma}_{w1}^{1/2}\hat{U}_b\hat{\Sigma}_b\hat{U}_b^T\tilde{\Sigma}_{w1}^{1/2} = \tilde{\Sigma}_{w1}^{1/2}\hat{U}_b(I+\hat{\Sigma}_b)\hat{U}_b^T\tilde{\Sigma}_{w1}^{1/2}$$

and a few of algebraic manipulations will give

$$\hat{S}_t^{-1}\hat{S}_b(\tilde{\Sigma}_{w1}^{-1/2}\hat{U}_b) = (\tilde{\Sigma}_{w1}^{-1/2}\hat{U}_b)((I+\hat{\Sigma}_b)^{-1}\hat{\Sigma}_b). \tag{40}$$

Hence the matrix Z can be obtained as $Z = \tilde{\Sigma}_{w1}^{-1/2}\hat{U}_{b1}$, which requires only the EVD in (39) and does not need to compute $\hat{S}_t^{-1}$. Eqs. (38) and (39) become

$$Z^T\tilde{U}_{w1}^T U_{t1}^T S_w U_{t1}\tilde{U}_{w1}Z = I \quad \text{and} \quad Z^T\tilde{U}_{w1}^T U_{t1}^T S_b U_{t1}\tilde{U}_{w1}Z = \hat{\Sigma}_{b1}$$

implying that the columns of $U_{t1}\tilde{U}_{w1}Z$ belong to null$(S_w)^c \cap$ null$(S_b)^c$. When two parts $U_{t1}\tilde{U}_{w2}Y$ and $U_{t1}\tilde{U}_{w1}Z$ are used for transformation matrix G_d, it will be better to normalize the columns in $U_{t1}\tilde{U}_{w1}Z$ so that effects of both parts can be balanced.

In this method, the role of PCA which is done by the transformation U_{t1} is to reduce the data dimension by removing the null space of S_t. Since we need only the basis of range(S_t), it can be found by the EVD of the smaller matrix $H_t^T H_t$ as in the efficient approach for LDA/GSVD. However, in fact even the EVD of $H_t^T H_t$ can be avoided. Since we need the basis of range(S_t), the reduced QR decomposition of $H_t = Q_1 R$ where columns in $Q_1 \in \mathbb{R}^{m\times n}$ are orthonormal and R is upper triangular will give the transformation matrix Q_1 which can replace U_{t1}. Since

$$\text{span}(U_{t1}) \subset \text{span}(Q_1),$$

any information is not lost by the transformation by Q_1, while the computational complexities can be saved by the reduced QR-decomposition instead of the EVD. If the difference between data dimension, m, and the number of data items, n, is not big as in the case exploited in Section 3.1, the PCA is not even necessary. We return to this point in Section 3 where nonlinear extensions of generalized LDA algorithms are introduced.

Relationship with the method *To-*$N(S_w)$

Recall from Section 2.3 that the method *To-*$N(S_w)$*)* projects the original space onto the null space of S_w using an orthonormal basis of null(S_w), and then in the projected space, a transformation that maximizes the between-class scatter is computed.

Since U_{t2} is a basis of null(S_t) and null$(S_t) \subset$ null(S_w), from (35)

$$\begin{bmatrix} U_{t1}^T \\ U_{t2}^T \end{bmatrix} S_w \begin{bmatrix} U_{t1} & U_{t2} \end{bmatrix} = \begin{bmatrix} \tilde{U}_w \tilde{\Sigma}_w \tilde{U}_w^T & 0 \\ 0 & 0 \end{bmatrix}. \tag{41}$$

By Eq. (41), we can obtain the EVD of S_w as

$$\begin{aligned} S_w &= \begin{bmatrix} U_{t1}\tilde{U}_w & U_{t2} \end{bmatrix} \begin{bmatrix} \tilde{\Sigma}_w & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{U}_w^T U_{t1}^T \\ U_{t2}^T \end{bmatrix} \\ &= \begin{bmatrix} U_{t1}\tilde{U}_{w1} & U_{t1}\tilde{U}_{w2} & U_{t2} \end{bmatrix} \begin{bmatrix} \tilde{\Sigma}_{w1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{U}_{w1}^T U_{t1}^T \\ \tilde{U}_{w2}^T U_{t1}^T \\ U_{t2}^T \end{bmatrix}. \end{aligned} \tag{42}$$

Eq. (42) shows that the columns of $V \equiv \begin{bmatrix} U_{t1}\tilde{U}_{w2} & U_{t2} \end{bmatrix}$ is the orthonormal basis of null(S_w). Hence the transformation by VV^T gives the projection onto the null space of S_w.

Now by the notation (36) and span$(U_{t2}) =$ null$(S_t) \subset$ null(S_b),

$$\begin{bmatrix} U_{t1}\tilde{U}_{w2} & U_{t2} \end{bmatrix} \begin{bmatrix} (U_{t1}\tilde{U}_{w2})^T \\ U_{t2}^T \end{bmatrix} S_b \begin{bmatrix} U_{t1}\tilde{U}_{w2} & U_{t2} \end{bmatrix} \begin{bmatrix} (U_{t1}\tilde{U}_{w2})^T \\ U_{t2}^T \end{bmatrix} = U_{t1}\tilde{U}_{w2}\bar{S}_b\tilde{U}_{w2}^T U_{t1}^T$$

which is the between-class scatter matrix in the projected space by VV^T. Let the EVD of $\bar{S}_b$ be

$$\bar{S}_b = [\bar{U}_{b1}\, \bar{U}_{b2}] \begin{bmatrix} \bar{\Sigma}_{b1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \bar{U}_{b1}^T \\ \bar{U}_{b2}^T \end{bmatrix}.$$

Then we have the transformation matrix G_e by the method *To-*$N(S_w)$*)* as

$$G_e = \begin{bmatrix} U_{t1}\tilde{U}_{w2} & U_{t2} \end{bmatrix} \begin{bmatrix} (U_{t1}\tilde{U}_{w2})^T \\ U_{t2}^T \end{bmatrix} U_{t1}\tilde{U}_{w2}\bar{U}_{b1} = U_{t1}\tilde{U}_{w2}\bar{U}_{b1} \tag{43}$$

which is exactly same as $U_{t1}\tilde{U}_{w2}Y$ in G_d of (37). Algorithm 2 summarizes the approach to compute G_e through Eq. (43). A main difference between the original algorithm for *To-*$N(S_w)$ and

Algorithm 2 An efficient algorithm for *To-*$N(S_w)$

1. Compute the EVD of S_t : $S_t = \begin{bmatrix} U_{t1} & U_{t2} \end{bmatrix} \begin{bmatrix} \Sigma_{t1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_{t1}^T \\ U_{t2}^T \end{bmatrix}$.
2. Compute the EVD of $\tilde{S}_w \equiv U_{t1}^T S_w U_{t1}$: $\tilde{S}_w = \begin{bmatrix} \tilde{U}_{w1} & \tilde{U}_{w2} \end{bmatrix} \begin{bmatrix} \tilde{\Sigma}_{w1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{U}_{w1}^T \\ \tilde{U}_{w2}^T \end{bmatrix}$.
3. Compute the SVD of $\bar{S}_b \equiv \tilde{U}_{w2}^T U_{t1}^T S_b U_{t1} \tilde{U}_{w2}$: $\bar{S}_b = \begin{bmatrix} \bar{U}_{b1} & \bar{U}_{b2} \end{bmatrix} \begin{bmatrix} \bar{\Sigma}_{b1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \bar{U}_{b1}^T \\ \bar{U}_{b2}^T \end{bmatrix}$.
4. $G_e = U_{t1} \tilde{U}_{w2} \bar{U}_{b1}$.

Algorithm 2 is in computational complexity. The original algorithm computes an orthonormal basis of the null space of S_w by the SVD of $H_w \in \mathbb{R}^{m \times n}$ or the EVD of $S_w \in \mathbb{R}^{m \times m}$ which requires $O(mn^2)$ or $O(m^3)$. On the other hand, Algorithm 2 computes an orthonormal basis of the null space of S_w via the range space of S_t which can be obtained by the EVD of $H_t^T H_t \in \mathbb{R}^{n \times n}$ at the cost of $O(n^3)$.

2.6 Analysis of Computational Complexities

In this section we analyze computational complexities for the methods discussed. For simplicity, let us assume that $H_b \in \mathbb{R}^{m \times r}$, $H_w \in \mathbb{R}^{m \times n}$ and $H_t \in \mathbb{R}^{m \times n}$ in (4-6) have already been computed, since the construction of scatter matrices is required in all the methods. Table 5 summarizes algorithms and main decompositions which are needed to compute the transformation matrix. For $H \in \mathbb{R}^{p \times q}$ and $p >> q$, when only eigenvectors corresponding to the nonzero eigenvalues of $HH^T \in \mathbb{R}^{p \times p}$ are needed, the algorithms in Table 5 utilize the approach of computing the EVD of $H^T H$ instead of HH^T as explained in Section 2.2. The computational complexity for the SVD decomposition depends on what parts need to be explicitly computed. We use flop counts for the analysis of computational complexities where one flop (floating point operation) represents roughly what is required to do one addition/subtraction or one multiplication/division [9]. For the

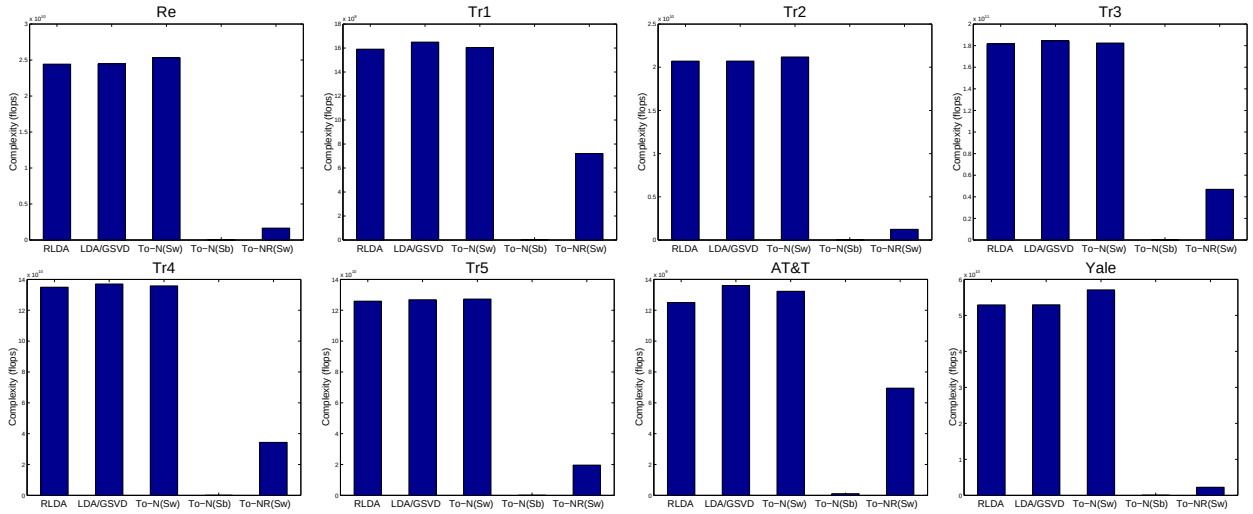

Figure 2: The figures compare computational complexities (flops) required according to the algorithms given in Table 5 using the sizes of training data used in experiments. For each dataset, from the left, RLDA, LDA/GSVD, *To*-$N(S_w)$, *To*-$R(S_b)$ and *To*-$NR(S_w)$.

SVD of a matrix $H \in \mathbb{R}^{p\times q}$ when $p >> q$,

$$H = U\Sigma V^T = [\underbrace{U_1}_{q}\ \underbrace{U_2}_{p-q}]\Sigma V^T,$$

where $U \in \mathbb{R}^{p\times p}$, $\Sigma \in \mathbb{R}^{p\times q}$ and $V \in \mathbb{R}^{q\times q}$, the complexities (flops) can be roughly estimated as follows [9, pp.254].

Required	Complexities
U_1, Σ	$6pq^2 + 11q^3$
U, Σ	$4p^2q + 13q^3$
U, Σ, V	$4p^2q + 22q^3$

For the multiplication of the $p_1 \times p_2$ matrix and the $p_2 \times p_3$ matrix, $2p_1p_2p_3$ flops were counted. For the simplicity, the rank was estimated as the number of data samples, n, or the number of classes, r, since it was close to either of two in most cases.

Figure 2 illustrates the total complexities corresponding to the algorithms in Table 5 for specific sizes of training datasets used in the experiments of the next section. The regularized LDA, LDA/GSVD [2] and the method *To*-$N(S_w)$ [4] have high computational complexities overall. The

method *To-*$R(S_b)$ [5] obtained the lowest computational costs compared with other methods and the method *To-*$NR(S_w)$ showed favorable computational complexities.

We showed that the transformation matrix G_e from the method *To-*$N(S_w)$*)* by Chen et al. is exactly same as the part which is obtained from the transformation to null(S_w) in *To-*$NR(S_w)$*)*. Then the computational complexity of the method *To-*$N(S_w)$*)* needs not to be among the highest, but it should be less costly than *To-*$NR(S_w)$*)*. The pseudocodes which implement Algorithm 1, Algorithm 2 and the approach utilizing Eq. (40) are given in Table 6. Figure 3 shows the comparison of complexities between original algorithms and the proposed ones for LDA/GSVD, *To-*$N(S_w)$ and *To-*$NR(S_w)$ respectively. As shown in Figure 3, the proposed algorithms for LDA/GSVD and *To-*$N(S_w)$ reduced the complexities of the original algorithms dramatically. These new algorithms can save computational complexities even more when the data dimension is much greater than the number of data. In the next section, we present experimental results which compare classification performances.

2.7 Experimental Comparisons of Generalized LDA Algorithms

In order to compare the discussed methods, we conducted extensive experiments using two types of datasets in text classification and face recognition.

Text classification is a task to assign a class label to a new document based on the information from pre-classified documents. A collection of documents are assumed to be represented as a term-document matrix, where each document is represented as a column vector and the components of the column vector denote frequencies of words appeared in the document. The term-document matrix is obtained after preprocessing with common words and rare term removal, stemming, term frequency and inverse term frequency weighting and normalization [10]. The term-document matrix representation often makes the high dimensionality inevitable. For all datasets[3], they were randomly split to the training set and the test set with the ratio of $4 : 1$ and experiments are repeated 10 times to reduce the possible variance caused from splitting data to training and test sets. After computing a transformation matrix using training data, both training data and test data were represented in the reduced dimensional space. In the transformed space, the nearest neighbor

[3]The text data sets were downloaded from http://www-users.cs.umn.edu/∼karypis/cluto/download.html, which were collected from Reuter-21578 and TREC-5, TREC-6, TREC-7 database.

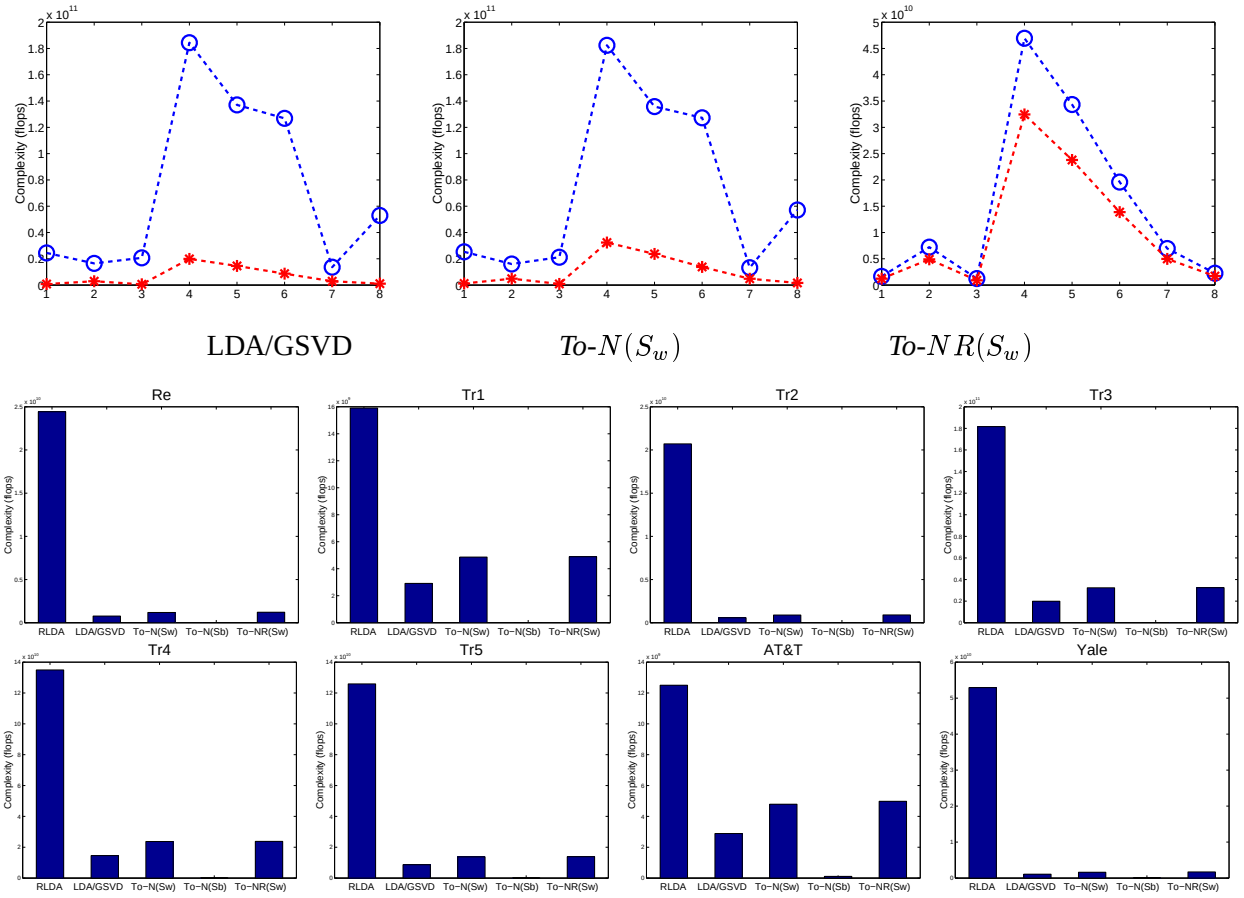

Figure 3: The figures in the first row compare complexities between original algorithms (o) and proposed ones (*) in Table 5 and 6, using the sizes of eight training data sets. The second and third rows: For each dataset, from the left, RLDA, Proposed LDA/GSVD, Proposed *To*-$N(S_w)$, *To*-$R(S_b)$ and Proposed *To*-$NR(S_w)$.

classifier was applied to compute the prediction accuracies for classification. For each data item in test set, it finds the nearest neighbor from the training data set and predicts a class label for the test data according to the class label of the nearest neighbor. Table 3 reports the mean prediction accuracies and standard deviation from 10 times random splitting to training and test sets.

The second experiment, face recognition, is a task to identify a person based on given face images with different facial expressions, illumination and poses. Since the number of pictures for each subject is limited and the data dimension is the number of pixels of a face image, face recognition data sets are typically severely undersampled. Our experiments used two datasets, AT&T (formerly ORL) face database and Yale face database. The AT&T database has 400 images, which

Data	Dim.	no. data	classes	RLDA	LDA/GSVD	*To*-$N(S_w)$	*To*-$R(S_b)$	*To*-$NR(S_w)$
Text Classification								
Re	3094	490	5	**95.8**	95.1	94.5	94.2	94.7
Tr1	5896	210	7	95.7	**98.3**	98.1	96.7	97.6
Tr2	5825	187	4	87.9	90.3	91.5	88.2	**91.8**
Tr3	8104	841	4	98.6	98.4	98.6	97.7	**98.7**
Tr4	7362	757	5	**98.0**	97.3	97.0	96.3	97.1
Tr5	8175	575	6	93.6	93.3	94.2	94.1	**94.4**
Face Recognition								
AT&T	2576	400	40	98.0	93.5	98.0	**99.0**	98.8
Yale	8586	165	15	97.6	**98.8**	97.6	89.7	98.2

Table 3: Prediction accuracies ($\%$). For RLDA, the best accuracy among $\alpha = 0.5, 1, 1.5$ is reported. For each dataset, the best prediction accuracy is shown in boldface.

consists of 10 images of 40 subjects. All the images were taken against a dark homogeneous background, with slightly varying lighting, facial expressions (open/closed eyes, smiling/non-smiling), and facial details (glasses/no-glasses). The subjects are in up-right, frontal positions with tolerance for some side movement [11]. For the manageable data sizes, the images have been downsampled from the size 92×112 to 46×56 by averaging the grey level values on 2×2 blocks. Yale face database contains 165 images, 11 images of 15 subjects. The 11 images per subject were taken under various facial expressions or configurations: center-light, with glasses, happy, left-light, without glasses, normal, right-light, sad, sleepy, surprised, and wink [12]. In our experiment, each image has been downsampled from 320×243 to 106×81 by averaging the grey values on 3×3 blocks. Since the number of images for each subject is small, leave-one-out method was performed where it takes one image for test set and the remaining images are used as a training set. Each image serves as a test datum by turns and the ratio of the number of correctly classified cases and the total number of data is considered as a prediction accuracy.

Table 3 shows the prediction accuracies in both experiments. For the regularized LDA, we report the best among the accuracies obtained with the regularization parameter $\alpha = 0.5, 1, 1.5$. Compared with the dramatic differences in computational complexities, no single method works

the best in all situations. The proposed fast algorithm for LDA/GSVD achieves competitive prediction accuracies while it requires low computational complexity. It is interesting to note that two datasets in face recognition gave the opposite results for LDA/GSVD and the method *To-*$R(S_b)$. In the experiment using AT&T database, the best prediction accuracy was achieved by the method *To-*$R(S_b)$, while LDA/GSVD obtained lowest prediction accuracy. Considering that the method *To-*$R(S_b)$ obtains all the projection vectors from null$(S_w)^c \cap$ null$(S_b)^c$, there might be overfitting caused by the projection using vectors from null$(S_w) \cap$ null$(S_b)^c$. As mentioned in the discussion of the relationship between LDA/GSVD and the method *To-*$N(S_w)$, when transformation matrices G_h by LDA/GSVD and G_e by *To-*$N(S_w)$ are both obtained from null$(S_w) \cap$ null$(S_b)^c$, we have

$$G_h^T S_b G_h = I_{r-1} \quad \text{and} \quad G_e^T S_b G_e = \tilde{\Sigma}_{b1}.$$

Through the experiments conducted, we observed that the dimension reduction by the transformation matrix $G_e \tilde{\Sigma}_{b1}^{-1/2}$ from

$$(\tilde{\Sigma}_{b1}^{-1/2})^T G_e^T S_b G_e \tilde{\Sigma}_{b1}^{-1/2} = I_{r-1}$$

gives the same prediction accuracies as LDA/GSVD.

3 Nonlinear Discriminant Analysis based on Kernel Methods

Linear dimension reduction is conceptually simple and has been used in many application areas. However, it has a limitation for the data which is not linearly separable since it is difficult to capture a nonlinear relationship with a linear mapping. In order to overcome such a limitation, nonlinear extensions of linear dimension reduction methods using kernel methods have been proposed [13, 14, 15, 16, 17]. The main idea of kernel methods is that without knowing the nonlinear feature mapping or the mapped feature space explicitly, we can work on the nonlinearly transformed feature space through kernel functions. It is based on the fact that for any kernel function κ satisfying Mercer's condition, there exists a reproducing kernel Hilbert space H and a feature map Φ such that

$$\kappa(x, y) = < \Phi(x), \Phi(y) > \tag{44}$$

where $< , >$ is an inner product in H [18, 19, 20].

Suppose that given a kernel function κ original data space is mapped to a feature space (possibly an infinite dimensional space) through a nonlinear feature mapping

$$\Phi : \mathcal{A} \subset \mathbb{R}^m \rightarrow \mathcal{F} \subset \mathbb{R}^N$$

satisfying (44). As long as the problem formulation depends only on the inner products between data points in $\mathcal{F}$ and not on the data points themselves, without explicit representation of the feature mapping Φ or the feature space $\mathcal{F}$, we can work on the feature space $\mathcal{F}$ through the relation (44). As positive definite kernel functions satisfying Mercer's condition, polynomial kernel

$$\kappa(x, y) = (\gamma_1(x \cdot y) + \gamma_2)^d, d > 0 \text{ and } \gamma_1, \gamma_2 \in \mathbb{R} \tag{45}$$

and Gaussian kernel

$$\kappa(x, y) = \exp(-\|x - y\|^2/2\sigma^2), \sigma \in \mathbb{R} \tag{46}$$

are in wide use.

We apply the kernel method to perform LDA in the feature space instead of the original input space. Given a kernel function κ, let $\mathcal{S}_b$ and $\mathcal{S}_w$ be the between-class and within-class scatter matrices in the feature space $\mathcal{F} \subset \mathbb{R}^N$ which has been transformed by a mapping Φ satisfying (44). Then the LDA in $\mathcal{F}$ finds a linear transformation

$$\mathcal{G} = [\varphi_1, \cdots, \varphi_l] \in \mathbb{R}^{N \times l},$$

where the columns of $\mathcal{G}$ are the generalized eigenvectors corresponding to the l largest eigenvalues of

$$\mathcal{S}_b \varphi = \lambda \mathcal{S}_w \varphi. \tag{47}$$

As in (4-5), $\mathcal{S}_b$ and $\mathcal{S}_w$ can be expressed as

$$\mathcal{S}_b = \mathcal{H}_b \mathcal{H}_b^T \quad \text{and} \quad \mathcal{S}_w = \mathcal{H}_w \mathcal{H}_w^T$$

where

$$\mathcal{H}_b = [\sqrt{n_1}(\tilde{c}_1 - \tilde{c}), \cdots, \sqrt{n_r}(\tilde{c}_r - \tilde{c})] \in \mathbb{R}^{N \times r}, \tag{48}$$

$$\mathcal{H}_w = [\Phi(A_1) - \tilde{c}_1 e_1, \cdots, \Phi(A_r) - \tilde{c}_r e_r] \in \mathbb{R}^{N \times n}, \tag{49}$$

$$\tilde{c}_i = \frac{1}{n_i} \sum_{j \in N_i} \Phi(a_j), \ \tilde{c} = \frac{1}{n} \sum_{i=1}^{n} \Phi(a_i) \ \text{ and } \ e_i = [1, \cdots, 1] \in \mathbb{R}^{1 \times n_i}.$$

The notation $\Phi(A_i)$ is used to denote

$$\Phi(A_i) = \Phi([a_j, \cdots, a_k]) = [\Phi(a_j), \cdots, \Phi(a_k)].$$

Note that any vector $\varphi \in \mathbb{R}^{N\times 1}$ can be represented as

$$\varphi = \varphi_1 + \varphi_2$$

where $\varphi_1 \in \text{span}\{\Phi(A)\}$ and $\varphi_2 \in \text{span}\{\Phi(A)\}^{\perp}$, and $\mathcal{S}_b\varphi_2 = 0$ and $\mathcal{S}_w\varphi_2 = 0$ for any $\varphi_2 \in \text{span}\{\Phi(A)\}^{\perp}$. Therefore, for any vector φ satisfying (47),

$$\mathcal{S}_b\varphi_1 = \mathcal{S}_b(\varphi_1 + \varphi_2) = \lambda S_w(\varphi_1 + \varphi_2) = \lambda S_w\varphi_1.$$

Hence we can restrict the solution space for (47) to $\text{span}\{\Phi(A)\}$. One may refer to [21] for an alternative explanation.

Let φ be represented as a linear combination of $\Phi(a_i)$, $i = 1, \cdots, n$,

$$\varphi = \sum_{i=1}^{n} u_i\Phi(a_i), \tag{50}$$

and define

$$\begin{aligned} &u = [u_1, \cdots, u_n]^T, \\ &\mathcal{K}_b = [\, b_{ij} \,]_{(1\le i\le n, 1\le j\le r)}, \quad b_{ij} = \sqrt{n_j}\quad \frac{1}{n_j}\sum_{p\in N_j}\kappa(a_i, a_p) - \frac{1}{n}\sum_{p=1}^{n}\kappa(a_i, a_p)\quad . \end{aligned} \tag{51}$$

Then

$$\mathcal{H}_b^T\varphi = \mathcal{K}_b^T u, \tag{52}$$

since

$$\begin{aligned} \mathcal{H}_b^T\varphi &= \begin{bmatrix} \sqrt{n_1}(\tilde{c}_1 - \tilde{c})^T \\ \vdots \\ \sqrt{n_r}(\tilde{c}_r - \tilde{c})^T \end{bmatrix} \left(\sum_{i=1}^{n} u_i\Phi(a_i) \right) \\ &= \begin{bmatrix} \sqrt{n_1}(\frac{1}{n_1}\sum_{p\in N_1}\Phi(a_p) - \frac{1}{n}\sum_{p=1}^{n}\Phi(a_p))^T \\ \vdots \\ \sqrt{n_r}(\frac{1}{n_r}\sum_{p\in N_r}\Phi(a_p) - \frac{1}{n}\sum_{p=1}^{n}\Phi(a_p))^T \end{bmatrix} \begin{bmatrix} \Phi(a_1), & \cdots, & \Phi(a_n) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix} \\ &= \mathcal{K}_b^T u. \end{aligned}$$

Similarly, we can obtain

$$\mathcal{H}_w^T \varphi = \mathcal{K}_w^T u \tag{53}$$

where

$$\begin{aligned} &\mathcal{K}_w = [\, w_{ij} \,]_{(1\le i\le n, 1\le j\le n)}\,, \\ &w_{ij} = \kappa(a_i, a_j) - \frac{1}{n_\delta} \sum_{p\in N_\delta} \kappa(a_i, a_p) \text{ when } a_j \text{ belongs to the class } \delta. \end{aligned} \tag{54}$$

From (52) and (53), for any $\varphi = \sum_{i=1}^{n} u_i \Phi(a_i)$ and $\psi = \sum_{i=1}^{n} v_i \Phi(a_i)$ we have

$$\begin{aligned} \mathcal{S}_b\varphi = \lambda \mathcal{S}_w \varphi \quad &\Leftrightarrow \quad \psi^T \mathcal{H}_b \mathcal{H}_b^T \varphi = \lambda \psi^T \mathcal{H}_w \mathcal{H}_w^T \varphi \\ &\Leftrightarrow \quad v^T \mathcal{K}_b \mathcal{K}_b^T u = \lambda v^T \mathcal{K}_w \mathcal{K}_w^T u \\ &\qquad\qquad \text{for } u = [u_1, \cdots, u_n]^T,\ v = [v_1, \cdots, v_n]^T \\ &\Leftrightarrow \quad \mathcal{K}_b \mathcal{K}_b^T u = \lambda \mathcal{K}_w \mathcal{K}_w^T u. \end{aligned} \tag{55}$$

Therefore, the generalized eigenvalue problem

$$\mathcal{S}_b\varphi = \lambda \mathcal{S}_w \varphi$$

becomes

$$\mathcal{K}_b \mathcal{K}_b^T u = \lambda \mathcal{K}_w \mathcal{K}_w^T u. \tag{56}$$

Note that $\mathcal{K}_b\mathcal{K}_b^T$ and $\mathcal{K}_w\mathcal{K}_w^T$ can be viewed as the between-class scatter matrix and within-class scatter matrix of the kernel matrix

$$K = [\, \kappa(a_i, a_j) \,]_{(1\le i\le n, 1\le j\le n)}$$

when each column in K is considered as a data point in the n-dimensional space. It can be observed by comparing the structures of $\mathcal{K}_b$ and $\mathcal{K}_w$ with those of $\mathcal{H}_b$ and $\mathcal{H}_w$ in (48-49). Since $\mathcal{K}_b\mathcal{K}_b^T$ and $\mathcal{K}_w\mathcal{K}_w^T$ are always singular, the classical LDA can not be applied in the feature space.

Considering each column

$$[\kappa(1, j), \cdots, \kappa(n, j)]^T \in \mathbb{R}^{n\times 1}$$

Algorithm 3 Nonlinear Discriminant Analysis

Given a data matrix $A = [a_1, \cdots, a_n] \in \mathbb{R}^{m \times n}$ with r classes and a kernel function κ, it computes the l dimensional representation of any input vector $z \in \mathbb{R}^{m \times 1}$ by applying the generalized LDA algorithm in the feature space mapped by the feature mapping Φ such that $\kappa(a_i, a_j) = < \Phi(a_i), \Phi(a_j) >$.

1. Compute $\mathcal{K}_b \in \mathbb{R}^{n \times r}$, $\mathcal{K}_w \in \mathbb{R}^{n \times n}$ and $\mathcal{K}_t \in \mathbb{R}^{n \times n}$ according to Eqs. (51), (54) and (57).
2. Compute transformation matrix $\mathcal{G}$ by applying the generalized LDA algorithms discussed in Section 2.
3. For any input vector $z \in \mathbb{R}^{m \times 1}$, a dimension reduced representation is computed by Eq. (59).

in the kernel matrix K as a data point in the n-dimensional space, we have the between-class, within-class and total scatter matrices as $\mathcal{K}_b\mathcal{K}_b^T$, $\mathcal{K}_w\mathcal{K}_w^T$ and $\mathcal{K}_t\mathcal{K}_t^T = \mathcal{K}_b\mathcal{K}_b^T + \mathcal{K}_w\mathcal{K}_w^T$, respectively. As in $\mathcal{K}_b$ and $\mathcal{K}_w$ of (51) and (54), $\mathcal{K}_t$ can be computed as

$$\mathcal{K}_t = [\, t_{ij} \,]_{(1 \le i \le n, 1 \le j \le n)}, \quad t_{ij} = \kappa(a_i, a_j) - \frac{1}{n} \sum_{p=1}^{n} \kappa(a_i, a_p). \tag{57}$$

Since $\mathcal{K}_b\mathcal{K}_b^T$ and $\mathcal{K}_w\mathcal{K}_w^T$ are both singular in the feature space, the classical LDA can not be applied. However, all the generalized LDA algorithms discussed in Section 2 are eligible for solving (56), resulting in nonlinear discriminant analysis algorithms. Let

$$\mathcal{G} = [u^{(1)}, \cdots, u^{(l)}] \in \mathbb{R}^{n \times l} \tag{58}$$

be the transformation matrix obtained by applying any generalized LDA algorithm in the feature space. Then the dimension reduced representation of z is given by

$$\mathcal{G}^T \begin{bmatrix} \kappa(a_1, z) \\ \vdots \\ \kappa(a_n, z) \end{bmatrix} \in \mathbb{R}^{l \times 1}. \tag{59}$$

Algorithm 3 summarizes nonlinear extensions of generalized LDA algorithms by kernel methods.

In the method *To-NR*(S_w), the purpose of the PCA step can be considered as a dimension reduction preserving the cluster structure in the data. From the EVD of S_t

$$S_t = \begin{bmatrix} U_{t1} & U_{t2} \end{bmatrix} \begin{bmatrix} \Sigma_{t1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_{t1}^T \\ U_{t2}^T \end{bmatrix},$$

the transformation by U_{t1}^T preserves distances between data points, since

$$\|a_i - a_j\|_2 = \left\| \begin{bmatrix} U_{t1}^T \\ U_{t2}^T \end{bmatrix} (a_i - a_j) \right\|_2 = \left\| \begin{bmatrix} U_{t1}^T (a_i - a_j) \\ 0 \end{bmatrix} \right\|_2 = \|U_{t1}^T (a_i - a_j)\|_2.$$

Therefore the between-class scatter and within-class scatter are also preserved in the space transformed by U_{t1}^T and the dimension of the transformed space becomes less than the number of data. However, in the feature space the effect of the PCA step is negligible unlike in undersampled problems, since the number of data and the data dimension are equal in the feature space and therefore the dimension reduction obtained by PCA is not effective compared with the cost for the SVD of the total scatter matrix. In the next section, the nonlinear discriminant analysis algorithms are compared experimentally and the computational complexities are analyzed.

3.1 Experimental Comparisons of Nonlinear Discriminant Analysis Algorithms

For this experiment, six datasets were used which were downloaded from UCI Machine Learning Repository. By randomly splitting the data to the training and test set of equal size and repeating it 10 times, ten pairs of training and test sets were constructed for each data. For the Bcancer and Bscale datasets, the ratio of training and test set was set as 4:1. Using the training set of the first pair among ten pairs and the nearest-neighbor classifier, 5 cross-validation was used in order to determine the optimal value for σ in the Gaussian kernel function

$$\kappa(x, y) = \exp \quad -\frac{\|x - y\|^2}{2\sigma^2} \quad .$$

After finding the optimal σ values, by running ten pairs of training and test sets mean prediction accuracies and standard deviations were calculated and they are reported in Table 4. In the regularization method, while the regularization parameter was set as 1, the optimal σ value was searched

					In the feature space				
Data	dim.	no.data	classes	LDA	RLDA	LDA/GSVD	*To-*$N(S_w)$	*To-*$R(S_b)$	*To-*$NR(S_w)$
Musk	166	6599	2	91.2	97.6	**99.4**	**99.4**	89.2	99.3
Isolet	617	7797	26	93.9	95.8	96.8	97.0	89.7	**97.1**
Car	6	1728	4	88.2	94.7	94.1	94.9	84.5	**95.2**
Mfeature[4]	649	2000	10	–	94.4	98.1	**98.3**	94.0	**98.3**
Bcancer	9	699	2	95.3	95.2	**96.4**	93.5	92.8	94.3
Bscale	4	625	3	87.0	**94.1**	86.5	86.5	86.5	86.1

Table 4: Prediction accuracies(%) by the classical LDA in the original space and the generalized LDA algorithms in the nonlinearly transformed feature space.

by the cross-validation. Table 4 also reports the prediction accuracies by the classical LDA in the original data space and it demonstrates that nonlinear discriminant analysis can improve prediction accuracies compared with linear discriminant analysis.

The next experiment is to compare performances of nonlinear discriminant analysis on undersampled problems, using the datasets in text classification and face recognition. The left figure in Figure 4 was obtained text data Tr5 and the right figure by AT&T face data. We note that results in other datasets showed similar tendency as in these two datasets. In each picture, the first group of five columns are by linear discriminant analysis in the original data space in the order of RLDA, LDA/GSVD, *To-*$N(S_w)$, *To-*$R(S_b)$ and *To-*$NR(S_w)$. The rest of groups of five columns were obtained by the Gaussian kernel function (46) and polynomial kernel function (45) of degree 2 and 3 respectively. Unlike the first experiment using datasets from UCI database, classification performances on undersampled problems were not much improved by nonlinear dimension reduction methods. It might indicate the linear separability in very high dimensional data.

Figure 5 illustrates the computational complexities using the specific sizes of the training data used in Table 4. Both in the original space and in the transformed feature space, the method *To-*$R(S_b)$ [4] gives the lowest computational complexities among the compared methods, while the performance is quite various depending on the data. Combining *To-*$R(S_b)$ with kernel methods does not make effective nonlinear dimension reduction method as shown in Table 4. In the generalized eigenvalue problem, $\mathcal{K}_b\mathcal{K}_b^T u = \lambda \mathcal{K}_w\mathcal{K}_w^T u$ where $\mathcal{K}_b\mathcal{K}_b^T, \mathcal{K}_w\mathcal{K}_w^T \in \mathbb{R}^{n\times n}$, the data dimension

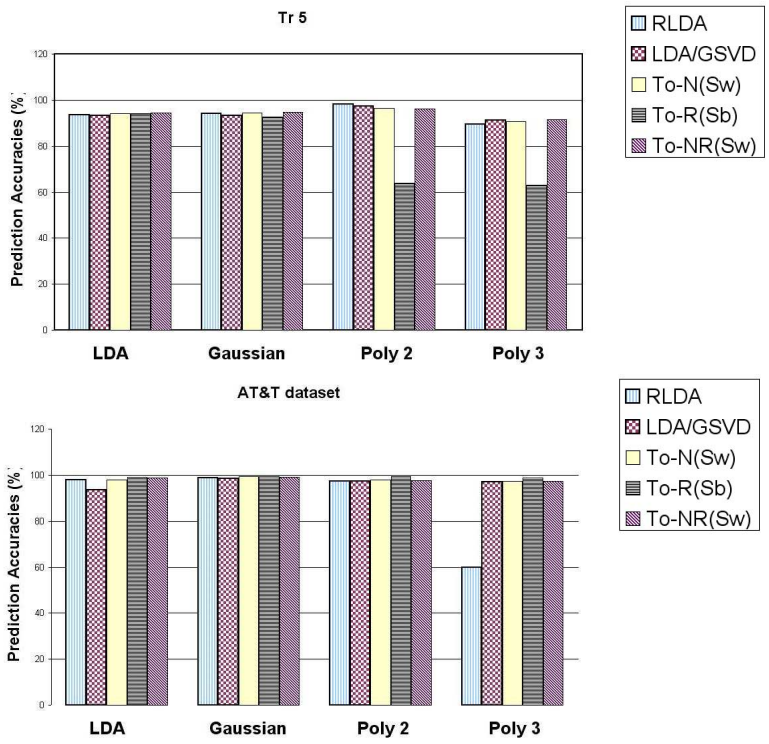

Figure 4: The figures compare classification performances by the generalized LDA algorithms in the original data space with those by nonlinear discriminant analysis.

is equal to the number of data and the rank of $\mathcal{K}_w\mathcal{K}_w^T$ is not severely smaller than the data dimension. However, poor performances by *To-*$R(S_b)$ demonstrate that the null space of $\mathcal{K}_w\mathcal{K}_w^T$ contains discriminative information. The regularized LDA obtained good performances both in the original space and in the feature space, while the computational complexity on the undersampled problems can be demanding. And also the regularization parameter should be determined experimentally. Figures 2 and 5 show that the proposed LDA/GSVD method can reduce greatly the computational cost of the original LDA/GSVD in both the original space and the feature space, while it can give comparable performances in both spaces. The method *To-*$NR(S_w)$ obtained high prediction accuracies. In the method *To-*$NR(S_w)$, through all the datasets we confirmed that without the PCA step it obtained the same prediction accuracies as with the PCA step while reducing the computational complexity greatly.

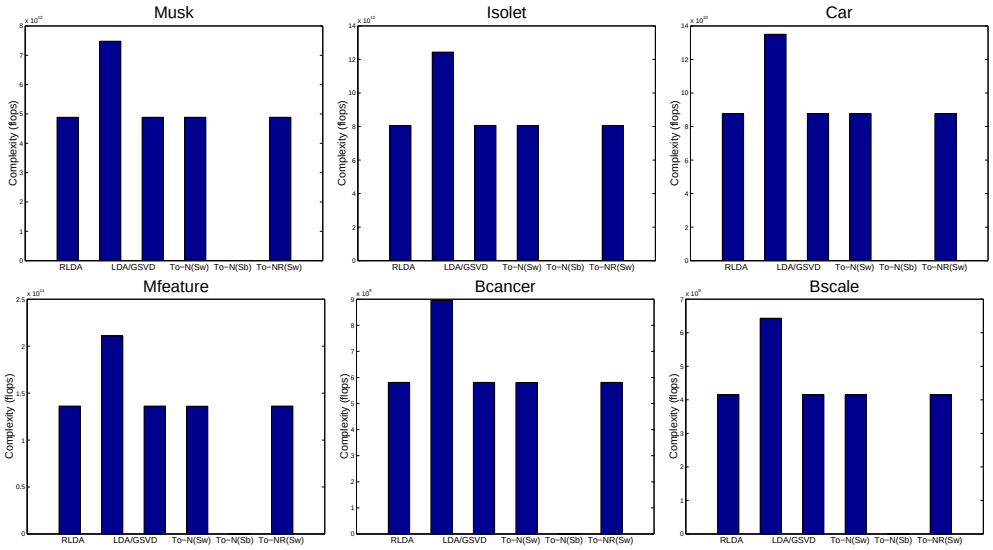

Figure 5: The figures compare complexities required for the generalized LDA algorithms in the feature space for specific problem sizes of training data used in Table 4. For each dataset, from the left, complexities of RLDA, LDA/GSVD, Proposed LDA/GSVD, *To*-$N(S_w)$, *To*-$R(S_b)$ and *To*-$NR(S_w)$ were shown.

4 Discussion

In this paper, we presented the theoretical and empirical comparisons of several generalized LDA algorithms proposed recently. By kernel methods and a formulation of generalized eigenvalue problem in the feature space, all the generalized LDA algorithms can be applied in the feature space, resulting in nonlinear feature extraction methods. Note that for two-class problems, RLDA in the feature space implements Kernel Fisher Discriminant (KFD) analysis by Mika et al.[14] which is a nonlinear extension of Fisher discriminant analysis. Through theoretical and experimental comparisons, we also proposed practical approaches for LDA/GSVD and the methods *To*-$N(S_w)$ and *To*-$NR(S_w)$ which can save computational complexities.

The conclusions can be summarized as follows.

1. Both in the original space and in the transformed feature space, the method *To*-$R(S_b)$ [4] gives the lowest computational complexities among the compared methods. If the computational complexity is a main concern, the method *To*-$R(S_b)$ can be recommended for undersampled problems. However, combining *To*-$R(S_b)$ with kernel methods does not make

effective nonlinear dimension reduction method as shown in Table 4.

2. The regularized LDA obtained good performances both in the original space and the feature space, while the computational complexity on the undersampled problems can be very demanding. And also the regularization parameter should be determined experimentally.

3. Figures 2 and 5 show that the proposed LDA/GSVD method can reduce greatly the computational cost of the original LDA/GSVD in both the original space and the feature space.

4. It was shown that the transformation matrix G_e from the method *To-*$N(S_w)$ by Chen et al. is exactly same as the part which is obtained from the transformation to null(S_w) in *To-*$NR(S_w)$. By utilizing this relationship, the computational complexity of the method *To-*$N(S_w)$ can be reduced less costly than *To-*$NR(S_w)$.

5. The method *To-*$NR(S_w)$ showed high prediction accuracies and quite low computational complexities both in original data space and in the nonlinearly transformed feature space. Note that the computational complexity of this method shown in Figure 5 was obtained without the PCA step and therefore much less than the one of the original method proposed in [6].

6. Experimental results using data sets from UCI database demonstrate that nonlinear discriminant analysis can improve prediction accuracies compared with linear discriminant analysis. Interestingly, classification performances on undersampled problems were not much improved by nonlinear dimension reduction methods. It might indicate the linear separability in very high dimensional data.

References

[1] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Acadamic Press, second edition, 1990.

[2] P. Howland, M. Jeon, and H. Park. Structure preserving dimension reduction for clustered text data based on the generalized singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 25(1):165–179, 2003.

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

[4] L. Chen, H.M. Liao, M. Ko, J. Lin, and G. Yu. A new LDA-based face recognition system which can solve the small sample size problem. *Pattern Recognition*, 33:1713–1726, 2000.

[5] H. Yu and J. Yang. A direct LDA algorithm for high-dimensional data- with application to face recognition. *Pattern Recognition*, 34:2067–2070, 2001.

[6] J. Yang and J.-Y. Yang. Why can LDA be performed in PCA transformed space? *Pattern Recognition*, 36:563–566, 2003.

[7] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley-interscience, New York, 2001.

[8] C.C. Paige and M.A. Saunders. Towards a generalized singular value decomposition. *SIAM Journal on Numerical Analysis*, 18:398–405, 1981.

[9] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, third edition, 1996.

[10] T.G. Kolda and D.P. O'Leary. A semidiscrete matrix decomposition for latent semantic indexing in information retrieval. *ACM transactions on Information Systems*, 16(4):322–346, 1998.

[11] http://www.uk.research.att.com/facedatabase.html.

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

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

[14] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller. Fisher discriminant analysis with kernels. In E.Wilson J.Larsen and S.Douglas, editors, *Neural networks for signal processing IX*, pages 41–48. IEEE, 1999.

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

[16] V. Roth and V. Steinhage. Nonlinear discriminant analysis using kernel functions. *Advances in neural information processing systems*, 12:568–574, 2000.

[17] S.A. Billings and K.L. Lee. Nonlinear fisher discriminant analysis using a minimum squared error cost function and the orthogonal least squares algorithm. *Neural networks*, 15(2):263–270, 2002.

[18] B. Schölkopf, S. Mika, C.J.C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A.J. Smola. Input space versus feature space in kernel-based methods. *IEEE transactions on neural networks*, 10(5):1000–1017, September 1999.

[19] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge, 2000.

[20] C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[21] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, A.J. Smola, and K.-R. Muller. Invariant feature extraction and classification in kernel spaces. *Advances in neural information processing systems*, 12:526–532, 2000.

RLDA: Regularized LDA

Compute the SVD of H_w : $H_w = U_w \Gamma_w V_w^T$. SVD($m \times n$)

Let $\mathbf{E_f} = U_w(\Gamma_w \Gamma_w^T + \alpha I)^{-1/2}$ and $\tilde{\mathbf{H}}_\mathbf{b} = E_f^T H_b$.

Compute the EVD of $\tilde{\mathbf{H}}_\mathbf{b}^\mathbf{T}\tilde{\mathbf{H}}_\mathbf{b}$: $\tilde{H}_b^T \tilde{H}_b = \tilde{U}_b \tilde{\Gamma}_b \tilde{U}_b^T$. EVD($r \times r$)

Let $\mathbf{W_f} = \tilde{H}_b(\tilde{U}_{b1}\tilde{\Gamma}_{b1}^{-1/2})$.

$\mathbf{G_f} = E_f W_f$.

LDA/GSVD

Compute the SVD of $Z = \begin{bmatrix} H_b^T \\ H_w^T \end{bmatrix}$: $Z = P \begin{bmatrix} \Lambda & 0 \\ 0 & 0 \end{bmatrix} U^T$, SVD($m \times (r+n)$)

where $s = \text{rank}(Z)$ and $\Lambda \in \mathbb{R}^{s \times s}$.

Compute the EVD of $\mathbf{P(1:r, 1:s)\,P(1:r, 1:s)^T}$: EVD($r \times r$)

$P(1:r, 1:s)\,P(1:r, 1:s)^T = W \Gamma\, W^T$.

Let $\mathbf{V_h} = P(1:r, 1:s)^T\,(W_1\,\Gamma_1^{-1/2})$.

$\mathbf{G_h} = U(:, 1:s)\,(\Lambda^{-1} V_h)$.

To-N(S_w): Projection to null(S_w)

Compute the SVD of H_w : $H_w = U_w \Gamma_w V_w^T$. SVD($m \times n$)

Let $\tilde{\mathbf{H}}_\mathbf{b} = U_{w2}(U_{w2}^T H_b)$.

Compute the EVD of $\tilde{\mathbf{H}}_\mathbf{b}^\mathbf{T}\tilde{\mathbf{H}}_\mathbf{b}$: $\tilde{H}_b^T \tilde{H}_b = \tilde{U}_b \tilde{\Gamma}_b \tilde{U}_b^T$. EVD($r \times r$)

Let $\mathbf{W_e} = \tilde{H}_b(\tilde{U}_{b1}\tilde{\Gamma}_{b1}^{-1/2})$.

$\mathbf{G_e} = U_{w2}(U_{w2}^T W_e)$.

To-R(S_b): Transformation to range(S_b)

Compute the EVD of $\mathbf{H_b^T H_b}$: $H_b^T H_b = U_b \Gamma_b U_b^T$. EVD($r \times r$)

Let $\mathbf{E_y} = H_b(U_{b1}\Gamma_{b1}^{-1})$ and $\tilde{\mathbf{H}}_\mathbf{w} = E_y^T H_w$.

Compute the EVD of $\tilde{\mathbf{H}}_\mathbf{w}\tilde{\mathbf{H}}_\mathbf{w}^\mathbf{T}$: $\tilde{H}_w \tilde{H}_w^T = \tilde{U}_w \tilde{\Gamma}_w \tilde{U}_w^T$. EVD($r \times r$)

$\mathbf{G_y} = E_y(\tilde{U}_{w1}\tilde{\Gamma}_{w1}^{-1/2})$.

To-NR(S_w): PCA plus Transformations to null(S_w) and range(S_w)	
Compute the EVD of $\mathbf{H_t^T H_t}$: $H_t^T H_t = U_t \Gamma_t U_t^T$.	EVD($n \times n$)
Let $\mathbf{E_d} = H_t(U_{t1}\Gamma_{t1}^{-1/2})$.	
Let $\mathbf{\tilde{H}_w} = E_d^T H_w$ and compute the SVD of $\tilde{H}_w$: $\tilde{H}_w = \tilde{U}_w \tilde{\Gamma}_w \tilde{V}_w^T$.	SVD($n \times n$)
Let $\mathbf{\bar{H}_b} = \tilde{U}_{w2}^T(E_d^T H_b)$, $\mathbf{\hat{H}_b} = \tilde{U}_{w1}^T(E_d^T H_b)$ and $\mathbf{\hat{H}_t} = \tilde{U}_{w1}^T(E_d^T H_t)$.	
Compute $\mathbf{\hat{S}_b} = \hat{H}_b \hat{H}_b^T$ and $\mathbf{\hat{S}_t} = \hat{H}_t \hat{H}_t^T$.	
Compute the EVD of $\mathbf{\bar{H}_b^T \bar{H}_b}$: $\bar{H}_b^T \bar{H}_b = \bar{U}_b \bar{\Gamma}_b \bar{U}_b^T$.	EVD($r \times r$)
Compute the eigenvectors U_1 with nonzero eigenvalues of $\mathbf{\hat{S}_t^{-1} \hat{S}_b}$.	EVD($n \times n$)
Let $\mathbf{G_{k1}} = E_d(\tilde{U}_{w2}(\bar{H}_b \bar{U}_{b1} \bar{\Gamma}_{b1}^{-1/2}))$ and $\mathbf{G_{k2}} = E_d(\tilde{U}_{w1} U_1)$.	
$G_k = [G_{k1}, G_{k2}]$ after normalization of G_{k2}	

Table 5: Multiplications needed are shown in boldface. In the EVDs, the subscript 1 was used to denote the block matrix corresponding to nonzero eigenvalues.

Proposed LDA/GSVD	
Compute the EVD of $\mathbf{H_t^T H_t}$: $H_t^T H_t = U_t \Gamma_t U_t^T$,	EVD($n \times n$)
Let $\mathbf{E_h} = H_t(U_{t1}\Gamma_{t1}^{-1})$ and $\tilde{\mathbf{H}}_\mathbf{b} = E_p^T H_b$.	
Compute the EVD of $\tilde{\mathbf{H}}_\mathbf{b}^\mathbf{T}\tilde{\mathbf{H}}_\mathbf{b}$: $\tilde{H}_b^T \tilde{H}_b = \tilde{U}_b \tilde{\Gamma}_b \tilde{U}_b^T$.	EVD($r \times r$)
Let $\mathbf{W_h} = \tilde{H}_b(\tilde{U}_{b1}\tilde{\Gamma}_{b1}^{-1/2})$.	
$\mathbf{G_h} = E_h W_h$.	
Proposed *To-N(S_w)*	
Compute the EVD of $\mathbf{H_t^T H_t}$: $H_t^T H_t = U_t \Gamma_t U_t^T$,	EVD($n \times n$)
Let $\mathbf{E_d} = H_t(U_{t1}\Gamma_{t1}^{-1/2})$.	
Let $\tilde{\mathbf{H}}_\mathbf{w} = E_d^T H_w$ and compute the SVD of $\tilde{H}_w$: $\tilde{H}_w = \tilde{U}_w \tilde{\Gamma}_w \tilde{V}_w^T$	SVD($n \times n$)
Let $\bar{\mathbf{H}}_\mathbf{b} = \tilde{U}_{w2}^T(E_d^T H_b)$	
Compute the EVD of $\bar{\mathbf{H}}_\mathbf{b}^\mathbf{T}\bar{\mathbf{H}}_\mathbf{b}$: $\bar{H}_b^T \bar{H}_b = \bar{U}_b \bar{\Gamma}_b \bar{U}_b^T$	EVD($r \times r$)
$\mathbf{G_e} = E_d(\tilde{U}_{w2}(\bar{H}_b \bar{U}_{b1}\bar{\Gamma}_{b1}^{-1/2}))$	
Proposed *To-NR(S_w)*	
Compute the EVD of $\mathbf{H_t^T H_t}$: $H_t^T H_t = U_t \Gamma_t U_t^T$,	EVD($n \times n$)
Let $\mathbf{E_d} = H_t(U_{t1}\Gamma_{t1}^{-1/2})$.	
Let $\tilde{\mathbf{H}}_\mathbf{w} = E_d^T H_w$ and compute the SVD of $\tilde{H}_w$: $\tilde{H}_w = \tilde{U}_w \tilde{\Gamma}_w \tilde{V}_w^T$	SVD($n \times n$)
Let $\bar{\mathbf{H}}_\mathbf{b} = \tilde{U}_{w2}^T(E_d^T H_b)$ and $\hat{\mathbf{H}}_\mathbf{b} = \tilde{\Gamma}_{w1}^{-1}\tilde{U}_{w1}^T(E_d^T H_b)$	
Compute the EVD of $\bar{\mathbf{H}}_\mathbf{b}^\mathbf{T}\bar{\mathbf{H}}_\mathbf{b}$: $\bar{H}_b^T \bar{H}_b = \bar{U}_b \bar{\Gamma}_b \bar{U}_b^T$	EVD($r \times r$)
Compute the EVD of $\hat{\mathbf{H}}_\mathbf{b}^\mathbf{T}\hat{\mathbf{H}}_\mathbf{b}$: $\hat{H}_b^T \hat{H}_b = \hat{U}_b \hat{\Gamma}_b \hat{U}_b^T$	EVD($r \times r$)
Let $\mathbf{G_{k1}} = E_d(\tilde{U}_{w2}(\bar{H}_b \bar{U}_{b1}\bar{\Gamma}_{b1}^{-1/2}))$ and $\mathbf{G_{k2}} = E_d(\tilde{U}_{w1}\tilde{\Gamma}_{w1}^{-1}(\hat{H}_b \hat{U}_{b1}\hat{\Gamma}_{b1}^{-1/2}))$.	
$G_k = [G_{k1}, G_{k2}]$ after normalization of G_{k2}.	

Table 6: The pseudocodes implement Algorithm 1, Algorithm 2 and the approach utilizing Eq. (40) for LDA/GSVD, *To-*$N(S_w)$ and *To-*$NR(S_w)$ respectively.