Kernel-based Fuzzy Feature Extraction Method and Its Application to Face Image Classification

Jinn-Min Yang[1], Pao-Ta Yu[2]

Department of Computer Science and Information Engineering, National Chung Cheng University
168 University Rd., Min-Hsiung 621, Chia-Yi, Taiwan, R.O.C.

[1]ygm@ms3.ntcu.edu.tw
[2]csipty@cs.ccu.edu.tw

***Abstract*— The Hughes phenomenon (or the curse of dimensionality) shows two essential directions for improving the classification performance on high-dimensional and small sample size (SSS) problems. One is to reduce the dimensionality of applied data by feature extraction or feature selection methods. The other is to increase the training sample size. In recent years some kernel-based feature extraction algorithms such as kernel principal component analysis (KPCA) have shown some appealing performances for various problems. These algorithms have the capabilities of handling extremely high-dimensional data and extracting nonlinear features. In this paper, an efficient nonparametric kernel-based feature extraction algorithm, namely kernel fuzzy feature extraction (KFFE), is proposed. In addition, two techniques, eigen-decomposition and regularization, are applied to the KFFE for mitigating the SSS problem. Two face databases are employed to confirm the effectiveness of the proposed algorithm. The experimental results demonstrate that the proposed KFFE obtains more satisfactory results as compared with some other kernel-based algorithms.**

***Keywords*—curse of dimensionality, feature extraction, dimension reduction, small sample size problem, face recognition, kernel method**

1. INTRODUCTION

Dimension reduction is an important scheme for high-dimensional classification problems, which aims to mitigate the Hughes phenomenon [1] (or the curse of dimensionality [2]) or other undesired properties of high-dimensional spaces so as to enhance classification performance. Linear discriminant analysis (LDA) [3] is one of the most well-known dimension reduction methods and has been successfully applied to many classification problems. The purpose of LDA is to find a linear transformation that can be used to project data from a high-dimensional space into a low-dimensional subspace with maximized class separability. The features extracted by LDA, however, are only linear that may fail for nonlinear problems. Besides, the LDA-like algorithms often suffer from handling extremely high-dimensional and small sample size (SSS) data, e.g., face image data. The SSS problem has significant influences on the performance of a pattern recognition system [4-6]. The face recognition problem is an illustration of extremely high-dimensional data with small sample size. For instance, a face image of size 64×64 pixels will be extended to a feature vector with 4096 dimensions. Then, the scatter matrices of classical LDA are of size 4096×4096, this will lead to a computational problem of handling such a big matrix. In other words, the drawback of general feature extraction methods is that the size of the matrix is proportional to the dimensionality of the data points.

In recent years, the kernel-based feature extraction algorithms are usually applied for overcoming the difficulties of handling extremely high-dimensional data and solving nonlinear problems simultaneously [7-9]. The study of kernel methods [10, 11] has gradually become an important theme due to the success of support vector machines (SVMs) [12] for pattern recognition in various fields. Generally speaking, any kernel method comprises two parts: a module that performs the mapping into the so-called embedding or feature space and a learning algorithm designed to discover linear patterns in that space [10]. There are two main fascinating properties of kernel-based algorithms in practical applications. First, the linear patterns can be represented efficiently via kernel trick [11, 12] without computing their coordinates explicitly; in other words, the algorithms can be implemented in terms of pairwise inner products in feature space and the inner products can be calculated

directly from the original data by employing a kernel function. Second, a linear relationship can be found in the feature space, which is equivalent to seeking the nonlinear relationship in the original space. From the aforementioned descriptions, the kernel-based algorithms can not only provide an alternative framework for finding the nonlinear relationship in original space but also reveal the importance of constructing new linear models in the future.

Two well-known kernel-based feature extraction methods, kernel principle component analysis (KPCA) [7] and generalized discriminant analysis (GDA) [8] (or kernel Fisher discriminant analysis; KFDA [13]), are the nonlinear extension of the classical PCA and LDA, respectively. They have been applied successfully in various problems of pattern recognition. As we know, the kernel-based feature extraction algorithms often encounter the SSS problem because the dimensionality of the feature space is extremely high or even infinite. Some techniques are involved to alleviate the SSS problem for kernel-based feature extraction models [8, 9, 13, 14]. KFDA [13] uses the technique of making the inner product matrix nonsingular by adding a multiple of the identity matrix. GDA [8] employs the QR decomposition technique to avoid the singularity by removing the zero eigenvalues. KDA/QR [9] and KDA/GSVD [14] are the nonlinear generalization of LDA/QR [15] and LDA/GSVD [16], respectively. LDA/QR is the LDA algorithm based on QR decomposition and LDA/GSVD is on generalized singular value decomposition (GSVD). Three categories for circumventing the singularity problem are described in [17].

In this paper, we propose a novel and efficient kernel-based feature extraction algorithm referred to as kernel fuzzy feature extraction (KFFE) which is the nonlinear extension of the fuzzy linear feature extraction (FLFE). For solving the SSS problem, two techniques are sequentially introduced to KFFE. The eigenvectors decomposition is first employed and the regularization technique is then taken. The effectiveness of the proposed KFFE is evaluated by means of two popular face recognition databases. The problem of face recognition is a typical case having small sample size and extremely high dimension, which is referred to as one of the most challenging applications in the pattern recognition. The experimental results exhibit the proposed KFFE provide a more stable and applicable framework for face recognition problem, as observed from a comparison with some other algorithms.

In the remainder of this paper, two popular kernel-based feature extraction methods, KPCA and GDA, are reviewed in Section 2. Then the proposed KFFE and its corresponding linear algorithm, FLFE, are described in detail in Section 3. Experimental designs and results are presented in Section 4 with conclusions given in Section 5.

2. KPCA AND GDA

Let Φ be a nonlinear mapping from the input space $\mathcal{R}^n$ into the feature space $\mathcal{F}$

$$\Phi : \mathcal{R}^n \to \mathcal{F}, \quad x \mapsto \Phi(x)$$

where the feature space $\mathcal{F}$ could have an arbitrarily large dimension or possibly infinite. Let $\Psi_{XY} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ denote a training set of observations, where the training sample $x_\ell \in \mathcal{R}^n$ and its corresponding label $y_\ell \in \mathcal{Y} = \{1, \ldots, L\}$ where L denotes the number of classes. Hence, the mapped dataset in feature space $\mathcal{F}$ is $\Psi_{\mathcal{X}Y} = \{(\Phi(x_1), y_1), \ldots, (\Phi(x_N), y_N)\}$. Suppose the number of training samples in class i is N_i , thus we have $\sum_{i=1}^{L} N_i = N$. Let $\kappa : \mathcal{R}^n \times \mathcal{R}^n \to \mathcal{R}$ denote the kernel function which is employed to express the computation of inner product of the mapped samples in $\mathcal{F}$, for example, $\kappa(x_i, x_j) = \Phi(x_i)^T \Phi(x_j) =< \Phi(x_i), \Phi(x_j) >$. In addition, $\mathcal{X}_i^T = [\Phi(x_1^{(i)}), \ldots, \Phi(x_{N_i}^{(i)})]$ and $\mathcal{X}^T = [\mathcal{X}_1^T, \ldots, \mathcal{X}_L^T]$ denote the transpose of data matrix of class i and overall data matrix in feature space $\mathcal{F}$, respectively. Then the kernel matrix $\mathcal{K}$ can be calculated by $\mathcal{X}\mathcal{X}^T$, i.e., $\mathcal{K} = \mathcal{X}\mathcal{X}^T = (\mathcal{K}_{pq})_{\substack{p=1,\ldots,L \\ q=1,\ldots,L}}$, where $\mathcal{K}_{pq} = \mathcal{X}_p \mathcal{X}_q^T = (\kappa_{ij})_{\substack{i=1,\ldots,N_p \\ j=1,\ldots,N_q}}$ and $\kappa_{ij} = \kappa(x_i, x_j)$. Two commonly used kernel functions are summarized as follows

$$\text{Polynomial} : \kappa(x, y) = (< x, y > +1)^d$$

$$\text{Gaussian RBF: } \kappa(x, y) = exp(-0.5\|x - y\|^2/\sigma^2)$$

where d and σ are the parameters of polynomial and Gaussian RBF kernels, respectively.

2.1. KPCA

KPCA performs classical PCA in feature space $\mathcal{F}$. For simplicity, the samples are assumed to be centered, that is, each sample is shifted by the global mean. Hence the covariance matrix can be constructed by

$$\mathcal{C} = \frac{1}{N}\sum_{\ell=1}^{N}\Phi(x_\ell)\Phi(x_\ell)^T = \frac{1}{N}\mathcal{X}^T\mathcal{X}. \quad (2.1)$$

The feature vectors are then computed by solving the following eigenvalue problem

$$\lambda v = \mathcal{C}v, \quad (2.2)$$

where λ and v are the eigenvalue and eigenvector of $\mathcal{C}$, respectively. Let (λ, v) denote the eigen-pair of $\mathcal{C}$. By the property of the reproducing kernels [11], the eigenvector v can be represented by a linear combination of all training samples in feature space $\mathcal{F}$, we then have

$$v = \sum_{\ell=1}^{N}\alpha_\ell\Phi(x_\ell) = \mathcal{X}^T\psi, \quad (2.3)$$

where ψ represents the column vector of expansion coefficients $\alpha_1, \ldots, \alpha_N$.

Combine (2.1) and (2.3) into (2.2) and multiply with $\mathcal{X}$ from the left, thus we obtain

$$N\lambda\mathcal{K}\psi = \mathcal{K}^2\psi. \quad (2.4)$$

To find ψ means solving the expansion coefficients $\alpha_1, \ldots, \alpha_N$ that can be obtained by solving the following eigenvalue problem

$$N\lambda\psi = \mathcal{K}\psi. \quad (2.5)$$

By the assumption that the training samples are centered, we have to substitute the kernel matrix $\mathcal{K}$ with

$$\hat{\mathcal{K}} = \mathcal{K} - \mathbf{1}_N\mathcal{K} - \mathcal{K}\mathbf{1}_N + \mathbf{1}_N\mathcal{K}\mathbf{1}_N, \quad (2.6)$$

where $(\mathbf{1}_N)_{pq} = 1/N$ for all p, q.

Let (λ_h, ψ_h) be the eigen-pair of $\hat{\mathcal{K}}$ and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m$. The orthonormal eigenvector v_h of $\mathcal{C}$ can be

$$v_h = \frac{1}{\sqrt{\lambda_h}}\mathcal{X}\psi_h, \quad h = 1, \ldots, m. \quad (2.7)$$

Finally, the hth component of a new pattern z with KPCA can be computed by projecting the mapped pattern $\Phi(z)$ onto v_h

$$< v_h, \Phi(z) >= \sum_{\ell=1}^{N}\alpha_{h\ell} < \Phi(x_\ell), \Phi(z) >= \sum_{\ell=1}^{N}\alpha_{h\ell}\kappa(x_\ell, z). \quad (2.8)$$

then the nonlinear principle components can be obtained through a kernel function κ, as shown from (2.8).

As is well known, the features extracted by PCA focus on discovering the most expressive but the most discriminating ones [18]; hence, KPCA remains the same. Consequently, the performance of KPCA is not so stable for classification tasks.

2.2. GDA

GDA extracts nonlinear discriminant features by performing LDA in the high dimensional feature space $\mathcal{F}$. The between-class scatter matrix $\mathcal{S}_b^{GDA}$, within-class scatter matrix $\mathcal{S}_w^{GDA}$ and total scatter matrix $\mathcal{S}_t^{GDA}$ of GDA are defined as

$$\mathcal{S}_b^{GDA} = \sum_{i=1}^{L} P_i(\mathbf{M}_i - \mathbf{M})(\mathbf{M}_i - \mathbf{M})^T, \quad (2.9)$$

$$\mathcal{S}_w^{GDA} = \sum_{i=1}^{L} P_i \sum_{\ell=1}^{N_i}(\Phi(x_\ell^{(i)}) - \mathbf{M}_i)(\Phi(x_\ell^{(i)}) - \mathbf{M}_i)^T, \quad (2.10)$$

$$\mathcal{S}_t^{GDA} = \mathcal{S}_b^{GDA} + \mathcal{S}_w^{GDA}, \quad (2.11)$$

where L is the number of classes, P_i is the prior probability of class i, $\mathbf{M}_i$ is the mean vector of class i and $\mathbf{M}$ is the global mean.

Since GDA performs LDA in feature space $\mathcal{F}$, it also aims to find a set of optimal eigenvectors by maximizing the Fisher criterion

$$J(\mathbf{A}) = \underset{\mathbf{A}}{\operatorname{argmax}} \frac{\mathbf{A}^T\mathcal{S}_b^{GDA}\mathbf{A}}{\mathbf{A}^T\mathcal{S}_t^{GDA}\mathbf{A}}. \quad (2.12)$$

The maximization of (2.12) is equivalent to the following generalized eigenvalue resolution

$$\mathcal{S}_b^{GDA}v_h = \lambda_h\mathcal{S}_t^{GDA}\ v_h, \quad (2.13)$$

where (λ_h, v_h) is the eigen-pair of $(\mathcal{S}_t^{GDA})^{-1}\mathcal{S}_b^{GDA}$ in the feature space $\mathcal{F}$ and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m$.

Then, the transformation matrix $\mathbf{A}$ is

$$\mathbf{A} = [v_1, \ldots, v_m].$$

As mentioned previously, the eigenvector v_h can be represented by a linear combination of all training samples in feature space $\mathcal{F}$

$$v_h = \sum_{i=1}^{L}\sum_{\ell=1}^{N_i}\alpha_{i\ell}\Phi(x_\ell^{(i)}) = \mathcal{X}^T\psi_h. \quad (2.14)$$

The training samples are assumed to be centered in feature space $\mathcal{F}$. Consequently, the $\mathcal{S}_b^{GDA}$ and the $\mathcal{S}_t^{GDA}$ can be derived as

$$\mathcal{S}_b^{GDA} = \frac{1}{N}\mathcal{X}^T\mathcal{W}\mathcal{X}, \quad (2.15)$$

$$\mathcal{S}_t^{GDA} = \frac{1}{N}\mathcal{X}^T\mathcal{X}, \quad (2.16)$$

where $\mathcal{W} = \operatorname{diag}(\mathbf{1}_{N_1}, \ldots, \mathbf{1}_{N_L})$ and $(\mathbf{1}_{N_i})_{pq} = 1/N_i$ for all p, q.

Applying (2.14)-(2.16) into (2.13) and multiply with $\mathcal{X}$ from the left, we can then obtain the following equation

$$\mathcal{K}\mathcal{W}\mathcal{K}\psi_h = \lambda\mathcal{K}\mathcal{K}\psi_h. \quad (2.17)$$

Here, $\mathcal{K}$ is the same as $\hat{\mathcal{K}}$ in (2.6).

Next, the eigenvectors decomposition of kernel matrix $\mathcal{K}$ is employed for stabilizing and improving the resolution in GDA.

Suppose $\mathcal{K} = P\Gamma P^T$, where P is an orthonormal matrix and Γ is the diagonal matrix of nonzero eigenvalues. Let $\beta_h = \Gamma P^T\psi_h$ then (2.17) can be converted to

$$P^T\mathcal{W}P\beta_h = \lambda\beta_h, \quad (2.18)$$

thus β_h is the eigenvector of $P^T\mathcal{W}P$. After β_h is obtained, the ψ_h can be computed by $P\Gamma^{-1}\beta_h$ and divided by $\sqrt{\psi_h^T\mathcal{K}\psi_h}$ to get normalized vectors v_h. Finally, the projection components of an unknown mapped sample $\Phi(z)$ can be expressed as

$$< v_h, \Phi(z) >= \sum_{i=1}^{L}\sum_{\ell=1}^{N_i}\alpha_{i\ell}\kappa(x_\ell^{(i)}, z), \text{ for } h = 1, \ldots, m \quad . \quad (2.19)$$

However, as the same with LDA, there are only $L-1$ features which can be extracted at most by using GDA.

3. KERNEL FUZZY FEATURE EXTRACTION

In this section, the idea of FLFE is briefly introduced and then the detailed deduction of KFFE is demonstrated.

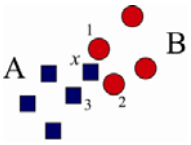
3.1 Fuzzy Linear Feature Extraction (FLFE)

The main idea of FLFE originates from the important finding of nonparametric discriminant analysis (NDA) [3, 19] which first exhibits an essential observation that the training samples approaching the class boundary should be emphasized and given more weights. The nonparametric weighted feature extraction (NWFE) [20] verifies the practicability of the important observation on hyperspectral image classification. Unlike NDA and NWFE, which implement the observation by using Euclidean distance directly, we discover the fuzzification procedure of the fuzzy *K*-nearest neighbor (F*K*NN) algorithm [21] that can be employed to find the samples near the class boundary more intuitively than NDA and NWFE. In addition, the membership values are introduced to the design of FLFE.

The fuzzification procedure of F*K*NN algorithm is

$$\mu_j(x_\ell^{(i)}) = \begin{cases} 0.51 + 0.49 \times \frac{n_j}{k} & \text{if } j = i \\ 0.49 \times \frac{n_j}{k} & \text{if } j \neq i \end{cases}, \quad (3.1)$$

where $x_\ell^{(i)}$ is the ℓth training sample in class i, n_j is the number of samples that belongs to class j, and k is a given constant which implies that k-nearest neighbors with respect to $x_\ell^{(i)}$ are included for computing the membership grades of $x_\ell^{(i)}$. In other words, we have $n_1 + n_2 + \cdots + n_L = k$. An example of two classes is illustrated in Fig. 1 and the parameter k is set to 3. From Eq. (3.1) we can find that the membership value $\mu_i(x_\ell^{(i)})$ is greater than or equal to 0.51 due to F*K*NN is a supervised learning method and the label of $x_\ell^{(i)}$ is already known. It is reasonable to give $\mu_i(x_\ell^{(i)})$ more than 0.50 even no neighbor (among k) is from the same class. The remaining 0.49 is shared according to the ratio of n_j to k. From (3.1), for $x_\ell^{(i)}$, a membership vector $\mu(x_\ell^{(i)}) = [\mu_1(x_\ell^{(i)}), \ldots, \mu_L(x_\ell^{(i)})]$ can be computed, in which each component denotes the grade with respect to each class. From the viewpoint of $x_\ell^{(i)}$, the bigger is the membership grade, the closer will be the class. Figure 2 illustrates the idea of applying the membership vector to find the samples approaching to the class boundary. The blue circled samples are regarded as those located near the class boundary.



$\mu(x) = [\mu_A, \mu_B]$
$\mu_A = 0.51 + 0.49 \times \frac{1}{3} = 0.673$
$\mu_B = 0.49 \times \frac{2}{3} = 0.327$

Fig. 1 Illustration of the computation of membership grades of sample *x* for $k = 3$.

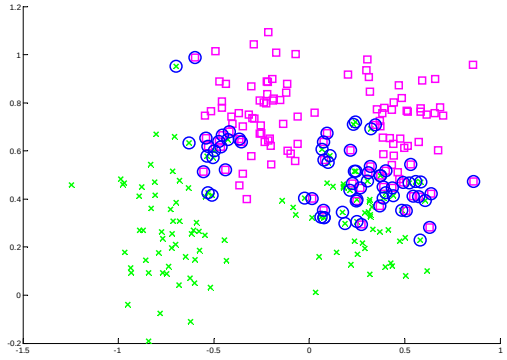


Fig. 2 Illustration of the boundary points detected by fuzzy membership.

The fuzzy within-class scatter matrix S_w^F and fuzzy between-class scatter matrix S_b^F are defined as follows.

$$S_w^F = \sum_{i=1}^{L} P_i \sum_{\ell=1}^{N_i} a_\ell^{(i,i)}(x_\ell^{(i)} - M_i(x_\ell^{(i)}))(x_\ell^{(i)} - M_i(x_\ell^{(i)}))^T \quad , \quad (3.2)$$

$$S_b^F = \sum_{i=1}^{L} P_i \sum_{\substack{j=1 \\ j\neq i}}^{L}\sum_{\ell=1}^{N_i} b_\ell^{(i,j)}(x_\ell^{(i)} - M_j(x_\ell^{(i)}))(x_\ell^{(i)} - M_j(x_\ell^{(i)}))^T \quad ,(3.3)$$

where P_i and $M_i(x_\ell^{(i)}) = \frac{1}{k}\sum_{s=1}^{k} x_{sNN}^{(i)}$ (or $M_j(x_\ell^{(i)})$) are the prior probability and the local mean of $x_\ell^{(i)}$ in class i (or j), respectively. $M_i(x_\ell^{(i)})$ is

computed by the k-nearest samples of $x_\ell^{(i)}$ in class i. In addition, $a_\ell^{(i,i)}$ and $b_\ell^{(i,j)}$ are the weighting factors of sample $x_\ell^{(i)}$ for within-class and between-class scatter matrices, respectively. They are defined as follows

$$a_\ell^{(i,i)} = 1 - [\mu_i(x_\ell^{(i)}) / \sum_{\ell=1}^{N_i} \mu_i(x_\ell^{(i)})],$$

$$b_\ell^{(i,j)} = \mu_j(x_\ell^{(i)}) / \sum_{\ell=1}^{N_i} \mu_j(x_\ell^{(i)}).$$

The designs of $a_\ell^{(i,i)}$ and $b_\ell^{(i,j)}$ confirm that the samples near class boundary can gain more weights. Unlike LDA, FLFE is nonparametric and is capable of extracting more than $L-1$ features, which is usually necessary for real data classification [6], [20].

3.2 Kernel Fuzzy Feature Extraction (KFFE)

Some key components must be taken into consideration before constructing the KFFE. The first one is about the distance measure in feature space $\mathcal{F}$ because the membership values have to be calculated through

$$\mu_j(\Phi(x_\ell^{(i)})) = \begin{cases} 0.51 + 0.49 \times \frac{n_j}{k} & \text{if } j = i \\ 0.49 \times \frac{n_j}{k} & \text{if } j \neq i \end{cases} \quad (3.4)$$

The second focal point is how the features are extracted by KFFE via kernel trick. Finally, we have to exhibit the projection of an unknown sample in feature space $\mathcal{F}$. In the following, we primarily reformulate the fuzzy within-class and between-class scatter matrices in feature space $\mathcal{F}$.

The fuzzy within-class scatter matrix in feature space $\mathcal{F}$, $\mathcal{S}_w^{KF}$, is defined as

$$\mathcal{S}_w^{KF} = \sum_{i=1}^{L} P_i \sum_{\ell=1}^{N_i} \mathbf{a}_\ell^{(i,i)} (\Phi(x_\ell^{(i)}) - \mathcal{M}_i(\Phi(x_\ell^{(i)}))) \cdot (\Phi(x_\ell^{(i)}) - \mathcal{M}_i(\Phi(x_\ell^{(i)}))^T, \quad (3.5)$$

where $\mathcal{M}_i(\Phi(x_\ell^{(i)})$ is the local mean of $\Phi(x_\ell^{(i)})$ in class i in feature space $\mathcal{F}$ and

$$\mathbf{a}_\ell^{(i,i)} = 1 - [\mu_i(\Phi(x_\ell^{(i)})) / \sum_{\ell=1}^{N_i} \mu_i(\Phi(x_\ell^{(i)})].$$

The fuzzy between-class scatter matrix in feature space $\mathcal{F}$, $\mathcal{S}_b^{KF}$, is given by

$$\mathcal{S}_b^{KF} = \sum_{i=1}^{L} P_i \sum_{\substack{j=1 \\ j \neq i}}^{L} \sum_{\ell=1}^{N_i} \mathbf{b}_\ell^{(i,j)} (\Phi(x_\ell^{(i)}) - \mathcal{M}_j(\Phi(x_\ell^{(i)}))) \cdot (\Phi(x_\ell^{(i)}) - \mathcal{M}_j(\Phi(x_\ell^{(i)}))^T, \quad (3.6)$$

where $\mathbf{b}_\ell^{(i,j)} = \mu_j(\Phi(x_\ell^{(i)})) / \sum_{\ell=1}^{N_i} \mu_j(\Phi(x_\ell^{(i)}))$.

For measuring the squared distance between samples x and y in feature space $\mathcal{F}$, the following formulation is used [11]:

$$\|\Phi(x) - \Phi(y)\| = \sqrt{\kappa(x,x) + \kappa(y,y) - 2\kappa(x,y)}, \quad (3.7)$$

where κ is a kernel function. From (3.7), we find that the distance can be directly computed via kernel function. In the following, we demonstrate how the features are extracted by KFFE via kernel trick.

Definition 1. *The local mean of $\Phi(x_\ell^{(i)})$ in calss j in feature space $\mathcal{F}$ is defined by*

$$\mathcal{M}_j(\Phi(x_\ell^{(i)})) = \frac{1}{k} \sum_{s=1}^{k} \Phi(x_{sNN}^{(j)}) = \mathcal{X}_j^T I_\ell^{(j)}, \quad (3.8)$$

where $\mathcal{X}_j^T = [\Phi(x_1^{(j)}), \ldots, \Phi(x_{N_j}^{(j)})]$ and $I_\ell^{(j)} = [i_1, \ldots, i_{N_j}]^T$, $\forall i_t \in \{\frac{1}{k}, 0\}, t = 1, \ldots, N_j$ and $\sum_{t=1}^{N_j} i_t = 1$.

Note that $I_\ell^{(j)}$ denotes a vector with components $1/k$ or 0, representing the contribution of each data point for computing $\mathcal{M}_j(\Phi(x_\ell^{(i)}))$.

Theorem 2. *The within-class scatter matrix can be expressed as*

$$\mathcal{S}_w^{KF} = \mathcal{X}^T \mathbf{W} \mathcal{X}, \quad (3.9)$$

where $\mathbf{W} = W_1 - W_2 - W_2^T + W_3$ with

$W_1 = \text{diag}(P_1 \Lambda_w^{(1,1)}, \ldots, P_L \Lambda_w^{(L,L)})$,

$W_2 = \text{diag}(P_1 \Lambda_w^{(1,1)} \mathbf{I}^{(1,1)}, \ldots, P_L \Lambda_w^{(L,L)} \mathbf{I}^{(L,L)})$,

$W_3 = \text{diag}(P_1 (\mathbf{I}^{(1,1)})^T \Lambda_w^{(1,1)} \mathbf{I}^{(1,1)}, \ldots, P_L (\mathbf{I}^{(L,L)})^T \Lambda_w^{(L,L)} \mathbf{I}^{(L,L)})$,

$\Lambda_w^{(i,i)} = \text{diag}(\mathbf{a}_1^{(i,i)}, \ldots, \mathbf{a}_{N_i}^{(i,i)})$ *and*

$\mathbf{I}^{(i,i)} = [I_1^{(i)}, \ldots, I_{N_i}^{(i)}]$.

Theorem 3. *The between-class scatter matrix can be expressed as*

$$\mathcal{S}_b^{KF} = \mathcal{X}^T (\mathbf{B} - \mathbf{W}_b) \mathcal{X} \quad (3.10)$$

where $\mathbf{B} = B_1 - B_2 - B_2^T + B_3$ and $\mathbf{W}_b = W_{b1} - W_{b2} - W_{b2}^T + W_{b3}$ with

$B_1 = \text{diag}(P_1 \sum_{j=1}^{L} \Lambda_b^{(1,j)}, \ldots, P_L \sum_{j=1}^{L} \Lambda_b^{(L,j)})$,

$$B_2 = \begin{bmatrix} P_1 \Lambda_b^{(1,1)} \boldsymbol{I}^{(1,1)} & \cdots & P_1 \Lambda_b^{(1,L)} \boldsymbol{I}^{(1,L)} \\ \vdots & \ddots & \vdots \\ P_L \Lambda_b^{(L,1)} \boldsymbol{I}^{(L,1)} & \cdots & P_L \Lambda_b^{(L,L)} \boldsymbol{I}^{(L,L)} \end{bmatrix},$$

$B_3 = \sum_{i=1}^{L} P_i \text{diag}((\boldsymbol{I}^{(i,1)})^T \Lambda_b^{(i,1)} \boldsymbol{I}^{(i,1)}, \ldots, (I^{(i,L)})^T \Lambda_b^{(i,L)} \boldsymbol{I}^{(L,L)})$

$\Lambda_b^{(i,j)} = \text{diag}(\mathbf{b}_1^{(i,j)}, \ldots, \mathbf{b}_{N_i}^{(i,j)})$, $\mathbf{I}^{(i,j)} = [I_1^{(j)}, \ldots, I_{N_i}^{(j)}]$

$W_{B1} = \text{diag}(P_1 \Lambda_b^{(1,1)}, \ldots, P_L \Lambda_b^{(L,L)})$,

$W_{B2} = \text{diag}(P_1 \Lambda_b^{(1,1)} \boldsymbol{I}^{(1,1)}, \ldots, P_L \Lambda_b^{(L,L)} \boldsymbol{I}^{(L,L)})$,

and

$W_{B3} = \text{diag}(P_1 (\boldsymbol{I}^{(1,1)})^T \Lambda_b^{(1,1)} \boldsymbol{I}^{(1,1)}, \ldots, P_L (\boldsymbol{I}^{(L,L)})^T \Lambda_b^{(L,L)} \boldsymbol{I}^{(L,L)})$

For similar proofs of the theorems, please refer to [22]. Now our goal is to find the transformation matrix $\mathbf{A}$ that maximizes the criterion

$$J(\mathbf{A}) = \underset{\mathbf{A}}{\operatorname{argmax}}\, tr((\mathbf{A}^T\mathcal{S}_w^{KF}\mathbf{A})^{-1}(\mathbf{A}^T\mathcal{S}_b^{KF}\mathbf{A})). \tag{3.11}$$

The maximization of (3.11) is equivalent to solve the generalized eigenvalue resolution

$$\mathcal{S}_b^{KF}v_h = \lambda_i\mathcal{S}_w^{KF}v_h, \tag{3.12}$$

where (λ_h, v_h) is the eigen-pair of $(\mathcal{S}_w^{KF})^{-1}\mathcal{S}_b^{KF}$ in the feature space $\mathcal{F}$ and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m$. Note that m is the dimension of the transformed space. Consequently, the transformation matrix $\mathbf{A}$ can be expressed as

$$\mathbf{A} = [v_1, \ldots, v_m].$$

Again, the eigenvector v_h lies in the span of all training samples in feature space $\mathcal{F}$, then we have

$$v_h = \mathcal{X}^T\psi_h. \tag{3.13}$$

Thus, the transformation matrix $\mathbf{A}$ can be represented as

$$\mathbf{A} = [v_1, \ldots, v_m] = \mathcal{X}^T[\psi_1, \ldots, \psi_m] = \mathcal{X}^T\Psi, \tag{3.14}$$

where $\Psi = [\psi_1, \ldots, \psi_m]$.

Therefore, from (3.9) to (3.14), to find $\mathbf{A}$ is equivalent to finding Ψ by

$$\Psi = \underset{\Psi}{\operatorname{argmax}}\, tr((\Psi^T\mathcal{K}\mathbf{W}\mathcal{K}\Psi)^{-1}(\Psi^T(\mathcal{K}(\mathbf{B}-\mathbf{W}_b)\mathcal{K}\Psi)). \tag{3.15}$$

Then ψ_h can be solved through the generalized eigenvalue problem

$$\lambda(\mathcal{K}\mathbf{W}\mathcal{K})\psi_h = \mathcal{K}(\mathbf{B}-\mathbf{W}_b)\mathcal{K}\psi_h. \tag{3.16}$$

From (3.16), we achieve the second goal that the feature ψ_h can be solved via kernel trick.

In the following, the same eigenvectors decomposition of the kernel matrix $\mathcal{K}$ is employed as in GDA. Again, let $\mathcal{K} = P\Gamma P^T$ and $\beta_h = \Gamma P^T\psi_h$, then β_h can be derived by solving the following generalized eigenvalue problem

$$\lambda(P^T\mathbf{W}P)\beta_h = P^T(\mathbf{B}-\mathbf{W}_b)P\beta_h, \tag{3.17}$$

Then β_h is the eigenvector of $(P^T\mathbf{W}P)^{-1}P^T(\mathbf{B}-\mathbf{W}_b)P$. However, the estimation of $(P^T\mathbf{W}P)^{-1}$ may be singular or nearly singular. The following regularization is therefore taken for circumventing the singularity problem

$$P^T\mathbf{W}P = (1-\mu)(P^T\mathbf{W}P) + \mu\text{diag}(P^T\mathbf{W}P), \tag{3.18}$$

where μ is a regularization parameter.

After β_h is found, the ψ_h can be computed by $P\Gamma^{-1}\beta_h$ and divided by $\sqrt{\psi_h^T\mathcal{K}\psi_h}$ to get normalized vectors v_h . We then obtain the transformation matrix $\mathbf{A}$.

Finally, the projected components of an unknown mapped sample $\Phi(z)$ in $\mathcal{F}$ can be calculated by

$$< v_h, \Phi(z) >= \sum_{i=1}^{L}\sum_{\ell=1}^{N_i}\alpha_{i\ell}\kappa(x_\ell^{(i)}, z), h = 1, \ldots, m. \tag{3.19}$$

Note that the projected components of an unknown sample are also computed by virtue of kernel trick.

The algorithm of KFFE is described as follows.

1. Calculate the membership values of each training sample of each class in feature space $\mathcal{F}$ by (3.4).
2. Compute $\Lambda_w^{(i,i)}$ and $\Lambda_b^{(i,j)}$.
3. Compute matrices $\mathbf{B}, \mathbf{W}_b, \mathbf{W}$ and kernel matrix $\mathcal{K}$.
4. Decompose $\mathcal{K}$ using eigenvectors decomposition, i.e., $\mathcal{K} = P\Gamma P^T$.
5. Estimate $P^T\mathbf{W}P$ by (3.18) and then compute β_h by (3.17), finally compute ψ_h and v_h.
6. Calculate the projection components of each sample by (3.19).

4. Experiment Design and Results

4.1. Data Set

In this study, two widely used face databases, AT&T[1] and UMIST[2], are employed to evaluate the effectiveness of the proposed KFFE. Each image in the two databases is represented as a vector with 10304 dimensions. Three other kernel-based feature extractions, GDA, KPCA and KDA/QR[3] with three kernel functions are included to investigate the performances on the two face databases. The three kernel functions are RBF kernel function and polynomial kernel function with degree 1 and 2, denoted as RBF, poly1 and poly2, respectively. Through this experiment, the classifier used is 1-nearest neighbor (1NN) and the number of features extracted by these algorithms is $L-1$. We randomly select p samples of each person from the databases for training and the rest for testing. The trials repeat 20 times for each case p and the

[1] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html.

[2] http://www.cs.toronto.edu/~roweis/data.html.

[3] http://www.cs.umn.edu/~jieping/Kernel.

average recognition accuracy of each algorithm is reported. Most of the results in this study are mainly based on STPRtool,[4] a toolbox build on MATLAB for pattern recognition, for obtaining reliable results. It implements a selection of statistical pattern recognition methods including GDA and KPCA.

The AT&T face database, formerly the ORL database of faces, consists of 10 different images of each of 40 distinct persons. These face images were taken between April 1992 and April 1994 at the Olivetti Research Laboratory. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position. The size of each image is pixels with a 256-level gray scale, which will be extended to a feature vector with 10304 dimensions. Some samples of a person in the database are demonstrated in Fig. 3. UMIST face database manually cropped by Dr. Daniel Graham is a multi-view dataset consisting of 575 images of 20 persons. The number of samples of each person is describes in Table 1. Each people covers a wide range of poses from profile to frontal views, some example images of one people are shown in Fig. 4. The size of each image is the same as that in AT&T database, so the dimensionality is 10304.

TABLE 1
THE NUMBER OF SAMPLES OF EACH PERSON IN UMIST DATABASE.

P	1	2	3	4	5	6	7	8	9	10
#	38	35	26	24	26	23	19	22	20	32
P	11	12	13	14	15	16	17	18	19	20
#	34	34	26	30	19	26	26	33	48	34

Fig. 3 Samples from the AT&T database.

4 http://cmp.felk.cvut.cz/cmp/software/stprtool/index.html/.

Fig. 4 Some samples of one person come from the UMIST database.

4.2. Experimental Results

The results on AT&T database are demonstrated in Fig. 5 and Table 2. Fig. 5 indicates the average recognition rate versus the subspace dimension for the four used algorithms, whereas the Gaussian RBF kernel is applied. In addition, the best average recognition rate versus the subspace dimension shown in the parentheses is listed in Table 2. From an overall viewpoint, the proposed KFFE obtains satisfactory results, no matter how many training samples are used and what kinds of kernel functions are applied. The recognition rate achieved by KDA/QR is competitive with KFFE when RBF kernel function is applied; however, it doesn't perform well when polynomial kernel functions are employed. The performance of KPCA is similar no matter what kernels are used on this database. In addition, GDA apparently has the worst performances than the other algorithms in most cases. To give more insight into these figures, the accuracy of KFFE increases very rapidly as the number of features increases, as compared with the other algorithms. In other words, KFFE achieves satisfactory results by only using fewer features and the results even over the best of some algorithms. The recognition rate of KFFE is 96.1% in the case $p = 5$, which is more than that of GDA in the case $p = 7$. For achieving a similar level of recognition rate, KFFE requires lower dimensions and smaller training samples, as is inferred from a comparison with the other algorithms. When polynomial kernel functions are applied, the differences of performance between KFFE, GDA and KDA/QR are larger than that of using RBF kernel functions.

The results on AT&T database are demonstrated in Fig. 6 and Table 3. Fig. 6 indicates the average recognition rate versus the subspace dimension for the four used algorithms,

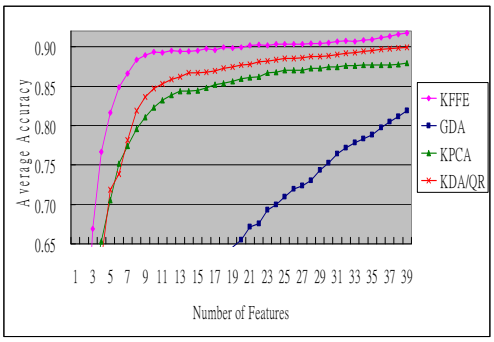
whereas the Gaussian RBF kernel is applied. Furthermore, the best accuracy versus the subspace dimension is summarized in Table 3. As the results demonstrated on AT&T database, the proposed KFFE obviously obtains more satisfactory results than the other algorithms, no matter how many training samples are used and what kind of kernel function in included. The differences of performance between KFFE and the other algorithms are more significant on this database than that on the AT&T database. KFFE outperforms the other algorithms more than 4% when p is less than 6 for all kernel functions. The recognition rate of KFFE increases notably than that of the other algorithms, which is similar to the performances on AT&T database. As observed from the line graph, when only 8 to 10 features are used, the results of KFFE are close to its own best accuracy, and over the best of the other algorithms. Especially, when the value of p is more than 6, an interesting observation is found that the number of used features corresponding to the best accuracy is about 10, that is , we can save the computational time of classification process to certain extent by using KFFE.

TABLE 2
THE BEST AVERAGE RECOGNITION RATE WITH THE SUBSPACE DIMENSION ON AT&T DATABASE.

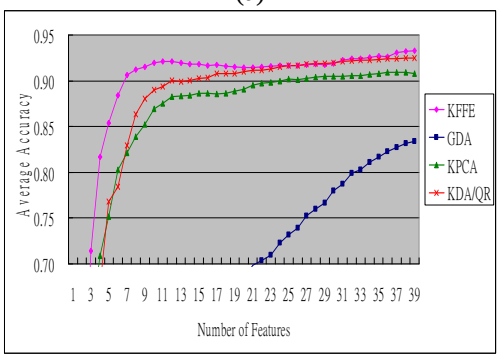
p	Kerf.	KFFE	GDA	KPCA	KDA/QR
3	RBF	0.918(39)	0.819(39)	0.880(39)	0.900(39)
	Poly1	0.917(39)	0.816(39)	0.880(39)	0.873(39)
	Poly2	0.916(39)	0.871(39)	0.871(33)	0.834(39)
4	RBF	0.933(39)	0.834(39)	0.909(36)	0.925(38)
	Poly1	0.933(39)	0.836(39)	0.910(37)	0.913(39)
	Poly2	0.935(39)	0.902(39)	0.903(35)	0.890(39)
5	RBF	0.961(39)	0.877(39)	0.939(38)	0.951(38)
	Poly1	0.961(39)	0.861(39)	0.939(38)	0.942(39)
	Poly2	0.961(39)	0.923(38)	0.929(39)	0.924(39)
6	RBF	0.972(23)	0.884(39)	0.962(39)	0.968(33)
	Poly1	0.972(23)	0.883(39)	0.962(39)	0.958(39)
	Poly2	0.974(39)	0.941(39)	0.956(39)	0.948(39)
7	RBF	0.977(33)	0.911(39)	0.966(30)	0.972(34)
	Poly1	0.978(33)	0.899(39)	0.966(30)	0.969(38)
	Poly2	0.980(39)	0.945(39)	0.960(24)	0.949(36)

TABLE 3
THE BEST AVERAGE RECOGNITION RATE WITH THE SUBSPACE DIMENSION ON UMIST DATABASE.

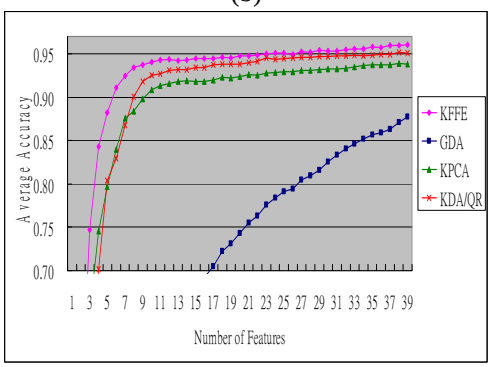
p	Kerf.	KFFE	GDA	KPCA	KDA/QR
3	RBF	0.822(19)	0.727(19)	0.695(18)	0.739(18)
	Poly1	0.822(19)	0.728(19)	0.695(18)	0.728(19)
	Poly2	0.805(19)	0.759(19)	0.694(18)	0.722 (19)
4	RBF	0.869(19)	0.793(19)	0.768(19)	0.805(19)
	Poly1	0.869(19)	0.792(19)	0.768(19)	0.811(19)
	Poly2	0.858(19)	0.823(19)	0.764(18)	0.792(18)
5	RBF	0.910(19)	0.837(19)	0.807(19)	0.847(14)
	Poly1	0.910(19)	0.828(19)	0.807(19)	0.854(19)
	Poly2	0.904(19)	0.867(19)	0.802(17)	0.845(18)
6	RBF	0.942(7)	0.888(19)	0.860(18)	0.899(16)
	Poly1	0.942(19)	0.891(19)	0.860(18)	0.905(19)
	Poly2	0.937(19)	0.910(19)	0.853(19)	0.896(15)
7	RBF	0.960(9)	0.911(19)	0.892(19)	0.924(15)
	Poly1	0.958(10)	0.910(19)	0.892(19)	0.931(19)
	Poly2	0.951(19)	0.928(19)	0.888(19)	0.917(17)



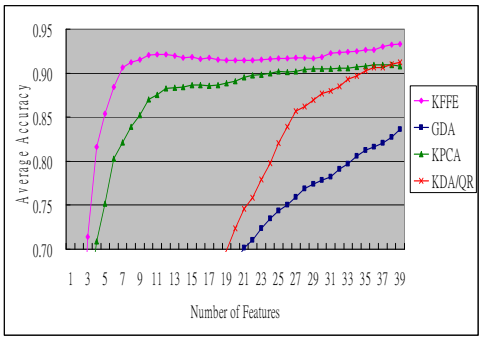
(a)



(b)

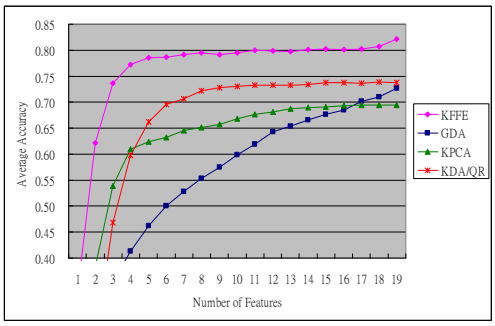


(c)

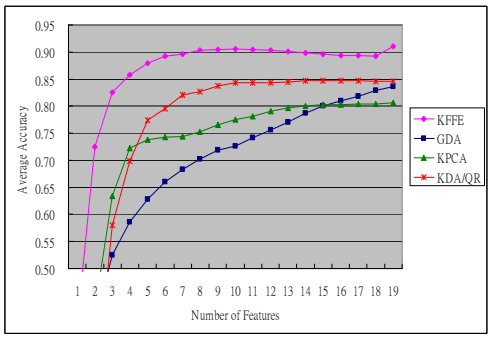


(d)

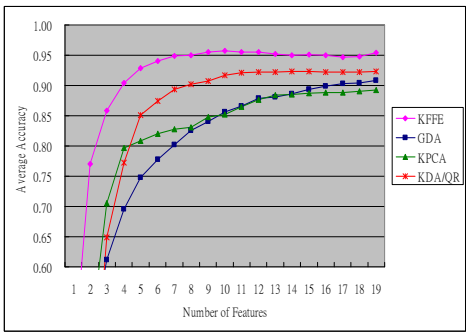
Fig. 5 (a) to (d) depict the best average recognition rate versus subspace dimension when applying the RBF kernel function on AT&T face dataset in the cases of $p = 3, 5, 7, \text{and } 9$, respectively.



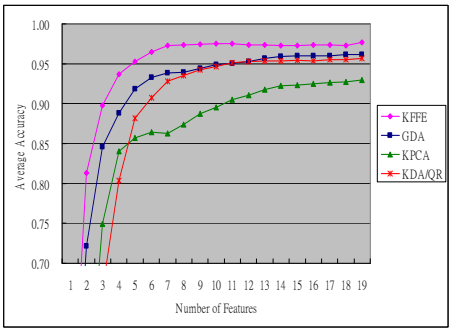
(a)



(b)



(c)



(d)

Fig. 6 (a) to (d) depict the average recognition rate versus the subspace dimension with the RBF kernel function on the UMIST database in the cases of $p = 3, 5, 7, \text{and } 9$, respectively.

5. CONCLUSIONS

In this paper, we proposed an efficient and applicable kernel-based feature extraction algorithm, KFFE, and evaluated its effectiveness on face recognition problem. In KFFE, the SSS problem was alleviated by introducing the eigen-decomposition and regularization techniques simultaneously. The experimental results showed that KFFE outperformed the other algorithms, GDA, KPCA and KDA/QR. The performance difference between KFFE and the other algorithms is particularly significant when using small training samples. Moreover, another essential advantage when applying KFFE to face recognition problem is that some computational time can be saved by scarifying only little classification accuracy

REFERENCES

[1] D.A. Landgrebe, Signal Theory Methods in Multispectral Remote Sensing, John Wiley and Sons, Hoboken, Chichester, 2003.

[2] R.O. Duda, P.E. Hart, and D.G. Stork, Pattern Classification, second ed., John Wiley & Sons, New York, 2001.

[3] K. Fukunaga, Introduction to Statistical Pattern Recognition, second ed., Academic Press, New York, 1990.

[4] S.J. Raudys, A.K. Jain, Small sample size effects in statistical pattern recognition: recommendations for practitioners, IEEE Trans. Pattern Anal. Mach. Intell, vol. 13, no. 3, 1991, pp.252-264.

[5] L.F. Chen, H.Y. M. Liao, C.C. Han and J.C. Lin, A new LDA based face recognition system which can solve the small sample size

problem, Pattern Recognition, vol. 33, 2000 pp.1713-1726.
[6] B.C. Kuo, K.Y. Chang, Feature extractions for small sample size classification problem, IEEE Trans. on Geoscience and Remote Sensing, vol. 45, no.3, 2007, pp.756-764.
[7] B. Schölkopf, A.J. Smola, K.R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Comput., vol. 10, 1998, pp.1299-1319.
[8] G. Baudat, F. Anouar, Generalized discriminant analysis using a kernel approach, Neural Comput., 12 (10) (2000) 2385–2404.
[9] T. Xiong, J. Ye, Q. Li, V. Cherkassky, R. Janardan, Efficient kernel discriminant analysis via QR decomposition, Proceedings of Advances in Neural Information Processing Systems, vol. 17, Cambridge, MA, , 2005, pp. 1529-1536.
[10] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, New York, 2004.
[11] B. Schölkopf, A.J. Smola. Learning with Kernels, The MIT Press, MA, 2002.
[12] V.N. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.
[13] S. Mika, G. Ratsch, J. Weston, Fisher discriminant analysis with kernels, Proc. Neural Networks for Signal Processing Workshop, Madison, WI, August 1999, pp. 41–48.
[14] C.H. Park, H. Park, Nonlinear discriminant analysis using kernel functions and the generalized singular value decomposition, SIAM J. Matrix Anal. Appl., 27 (1), (2005) 87-102.
[15] J. Ye, Q. Li, LDA/QR: an efficient and effective dimension reduction algorithm and its theoretical foundation, Pattern Recognition, 37 (4) (2004) 851-854.
[16] P. Howland, M. Jeon, H. Park, Structure preserving dimension reduction for clustered text data based on the generalized singular value decomposition, SIAM J. Matrix Anal. Appl., 25 (1), (2003) 165-179.
[17] A.R. Webb, Statistical Pattern Recognition, second ed., New Jersey, John Wiley & Sons Inc, 2002.
[18]M.H. Yang, Kernel eigenfaces vs. kernel Fisherfaces: face recognition using kernel methods, Proceedings of the Fifth International Conference on Automatic Face and Gesture Recognition , Washington, DC, USA, 2002, pp.215–220.
[19] K. Fukunaga, J.M. Mantock, Nonparametric discriminant analysis, IEEE Trans. Pattern Anal. Mach. Intell. 5 (1983) 671-678.
[20] B.C. Kuo, D.A. Landgrebe, Nonparametric weighted feature extraction for classification, IEEE Transaction on Geoscience and Remote Sensing 42 (5) (2004) 1096-1105.
[21] J.M. Keller, M.R. Gray, J.A. Givens, A fuzzy k-nearest neighbor algorithm, IEEE Transaction on Systems, Man, and Cybernetics 15 (4) (1985) 580-58.
[22] B. C. Kuo, C. H. Li, and J. M. Yang, Kernel Nonparametric Weighted Feature Extraction for Hyperspectral Image Classification, IEEE Transaction on Geoscience and Remote Sensing, vol. 47, no. 4, pp. 1139-1155, 2008.