Face Recognition Using a Kernel Fractional-Step Discriminant Analysis Algorithm

Guang Dai [a], Dit-Yan Yeung [a] & Yun-Tao Qian [b]

[a] *Department of Computer Science and Engineering*
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong

[b] *College of Computer Science*
Zhejiang University
Hangzhou, 310027, P.R. China

Abstract

Feature extraction is among the most important problems in face recognition systems. In this paper, we propose an enhanced kernel discriminant analysis (KDA) algorithm called *kernel fractional-step discriminant analysis* (KFDA) for nonlinear feature extraction and dimensionality reduction. Not only can this new algorithm, like other kernel methods, deal with nonlinearity required for many face recognition tasks, it can also outperform traditional KDA algorithms in resisting the adverse effects due to outlier classes. Moreover, to further strengthen the overall performance of KDA algorithms for face recognition, we propose two new kernel functions: cosine fractional-power polynomial kernel and non-normal Gaussian RBF kernel. We perform extensive comparative studies based on the YaleB and FERET face databases. Experimental results show that our KFDA algorithm outperforms traditional kernel principal component analysis (KPCA) and KDA algorithms. Moreover, further improvement can be obtained when the two new kernel functions are used.

Key words: Face recognition, feature extraction, nonlinear dimensionality reduction, kernel discriminant analysis, kernel fractional-step discriminant analysis.

1 Introduction

1.1 Linear Discriminant Analysis

Linear subspace techniques have played a crucial role in face recognition research for linear dimensionality reduction and feature extraction. The two most well-known methods are principal component analysis (PCA) and linear discriminant analysis (LDA), which are for feature extraction under the unsupervised and supervised learning settings, respectively. Eigenface [27], one of the most successful face recognition methods, is based on PCA. It finds the optimal projection directions that maximally preserve the data variance. However, since it does not take into account class label information, the "optimal" projection directions found, though useful for data representation and reconstruction, may not give the most discriminating features for separating different face classes. On the other hand, LDA seeks the optimal projection directions that maximize the ratio of between-class scatter to within-class scatter. Since the face image space is typically of high dimensionality but the number of face images available for training is usually rather small, a major computational problem with the LDA algorithm is that the within-class scatter matrix is singular and hence the original LDA algorithm cannot be applied directly. This problem can be attributed to undersampling of data in the high-dimensional image space.

Over the past decade, many variants of the original LDA algorithm have been proposed for face recognition, with most of them trying to overcome the problem due to undersampling. Some of these methods perform PCA first before applying LDA in the PCA-based subspace, as is done in Fisherface (also known as PCA+LDA) [2,25]. Belhumeur *et al.* [2] carried out comparative experiments based on the Harvard and Yale face databases and found that Fisherface did give better performance than Eigenface in many cases. However, by analyzing the sensitivity on the spectral range of the within-class eigenvalues, Liu and Wechsler [12] found that the generalization ability of Fisherface can be degraded since some principal components with small eigenvalues correspond to high-frequency components and hence can play the role of latent noise in Fisherface. To overcome this limitation, Liu and Wechsler [12] developed two enhanced LDA models for face recognition by simultaneous diagonalization of the within-class and between-class scatter matrices instead of the conventional LDA procedure. More recently, some other LDA-based methods have been developed for face recognition on the basis of different views. Ye *et al.* proposed LDA/GSVD [33] and LDA/QR [34] by employing generalized singular value decomposition (GSVD) and QR decomposition, respectively, to solve a generalized eigenvalue problem. Some other researchers have proposed the direct LDA algorithm and variants [3,4,30,35]. These methods have been

found to be both efficient and effective for many face recognition tasks.

For multi-class classification problems involving more than two classes, a major drawback of LDA is that the conventional optimality criteria defined based on the scatter matrices do not correspond directly to classification accuracy [10,16,23]. An immediate implication is that optimizing these criteria does not necessarily lead to an increase in the accuracy. This phenomenon can also be explained in terms of the adverse effects due to the so-called outlier classes [15], resulting in inaccurate estimation of the between-class scatter. As a consequence, the linear transformation of traditional LDA tends to overemphasize the inter-class distances between already well-separated classes in the input space at the expense of classes that are close to each other leading to significant overlap between them. To tackle this problem, some outlier-class-resistant schemes [10,15,23] based on certain statistical model assumptions have been proposed. They are common in that a weighting function is incorporated into the Fisher criterion by giving higher weights to classes that are closer together in the input space as they are more likely to lead to misclassification. Although these methods are generally more effective than traditional LDA, it is difficult to set the weights appropriately particularly when the statistical model assumptions may not be valid in such applications as face recognition where the data are seriously undersampled. Recently, Lotlikar and Kothari [16] proposed an interesting idea which allows fractional steps to be made in dimensionality reduction. The method, referred to as fractional-step LDA (FLDA), allows for the relevant distances to be more correctly weighted. A more recent two-phase method, called direct FLDA (DFLDA) [18], attempts to apply FLDA to high-dimensional face patterns by combining the FLDA and direct LDA algorithms.

1.2 Kernel Discriminant Analysis

In spite of their simplicity, linear dimensionality reduction methods have limitations under situations when the decision boundaries between classes are nonlinear. For example, in face recognition applications where there exists high variability in the facial features such as illumination, facial expression and pose, high nonlinearity is commonly incurred. This calls for nonlinear extensions of conventional linear methods to deal with such situations. The past decade has witnessed the emergence of a powerful approach in machine learning called kernel methods, which exploit the so-called "kernel trick" to devise nonlinear generalizations of linear methods while preserving the computational tractability of their linear counterparts. The first kernel method proposed is support vector machine (SVM) [28], which essentially constructs a separating hyperplane in the high-dimensional (possibly infinite-dimensional) feature space $\mathcal{F}$ obtained through a nonlinear feature map $\phi : \mathbb{R}^n \to \mathcal{F}$. The

kernel trick allows inner products in the feature space to be computed entirely in the input space without performing the mapping explicitly. Thus, for linear methods which can represent the relationships between data in terms of inner products only, they can readily be "kernelized" to give their nonlinear extensions. Inspired by the success of SVM, kernel subspace analysis techniques have been proposed to extend linear subspace analysis techniques to nonlinear ones by applying the same kernel trick, leading to performance improvement in face recognition over their linear counterparts. As in other kernel methods, these kernel subspace analysis methods essentially map each input data point $x \in \mathbb{R}^n$ into some feature space $\mathcal{F}$ via a mapping ϕ and then perform the corresponding linear subspace analysis in $\mathcal{F}$. Schölkopf *et al.* [24] have pioneered to combine the kernel trick with PCA to develop the kernel PCA (KPCA) algorithm for nonlinear principal component analysis. Based on KPCA, Yang *et al.* [32] proposed a kernel extension to the Eigenface method for face recognition. Using a cubic polynomial kernel, they showed that kernel Eigenface outperforms the original Eigenface method. Moghaddam [21] also demonstrated that KPCA using the Gaussian RBF kernel gives better performance than PCA for face recognition. More recently, Liu [11] further extended kernel Eigenface to include fractional-power polynomial models that correspond to non-positive semi-definite kernel matrices.

In the same spirit as the kernel extension of PCA, kernel extension of LDA, called kernel discriminant analysis (KDA), has also been developed and found to be more effective than PCA, KPCA and LDA for many classification applications due to its ability in extracting nonlinear features that exhibit high class separability. Mika *et al.* [20] first proposed a two-class KDA algorithm, which was later generalized by Baudat and Anouar to give the generalized discriminant analysis (GDA) algorithm [1] for multi-class problems. Motivated by the success of Fisherface and KDA, Yang [31] proposed a combined method called kernel Fisherface for face recognition. Liu *et al.* [13] conducted a comparative study on PCA, KPCA, LDA and KDA and showed that KDA outperforms the other methods for face recognition tasks involving variations in pose and illumination. Other researchers have proposed a number of KDA-based algorithms [6,14,17,19,29] addressing different problems in face recognition applications.

1.3 This Paper

However, similar to the case of LDA-based methods, the adverse effects due to outlier classes also affect the performance of KDA-based algorithms. In fact, if the mapped data points in $\mathcal{F}$ are more separated from each other, such effects can become even more significant. To remedy this problem, we propose in this paper an enhanced KDA algorithm called kernel fractional-step discrimi-

nant analysis (KFDA). The proposed method, similar to DFLDA, involves two steps. In the first step, KDA with a weighting function incorporated is performed to obtain a low-dimensional subspace. In the second step, a subsequent FLDA procedure is applied to the low-dimensional subspace to accurately adjust the weights in the weighting function through making fractional steps, leading to a set of enhanced features for face recognition.

Moreover, recent research in kernel methods shows that an appropriate choice of the kernel function plays a crucial role in the performance delivered. To further improve the performance of KDA-based methods for face recognition, we propose two new kernel functions called cosine fractional-power polynomial kernel and non-normal Gaussian RBF kernel. Extensive comparative studies performed on the YaleB and FERET face databases give the following findings:

(1) Compared with other methods such as KPCA and KDA, KFDA is much less sensitive to the adverse effects due to outlier classes and hence is superior to them in terms of recognition accuracy.
(2) For both KDA and KFDA, the two new kernels generally outperform the conventional kernels, such as polynomial kernel and Gaussian RBF kernel, that have been commonly used in face recognition. KFDA, when used with the new kernels, delivers the highest face recognition accuracy.

The rest of this paper is organized as follows. In Section 2, we briefly review the conventional KPCA and KDA algorithms. In Section 3, the new KFDA algorithm is presented, and then the two new kernel functions are introduced to further improve the performance. Extensive experiments have been performed with results given and discussed in Section 4, demonstrating the effectiveness of both the KFDA algorithm and the new kernel functions. Finally, Section 5 concludes this paper.

2 Brief Review of Kernel Principal Component Analysis and Kernel Discriminant Analysis

The key ideas behind kernel subspace analysis methods are to first implicitly map the original input data points into a feature space $\mathcal{F}$ via a feature map ϕ and then implement some linear subspace analysis methods with the corresponding optimality criteria in $\mathcal{F}$ to discover the optimal nonlinear features with respect to the criteria. Moreover, instead of performing the computation for the linear subspace analysis directly in $\mathcal{F}$, the implementation based on the kernel trick is completely dependent on a kernel matrix or Gram matrix whose entries can be computed entirely from the input data points without requiring the corresponding feature points in $\mathcal{F}$.

Let $\mathcal{X}$ denote a training set of N face images belonging to c classes, with each image represented as a vector in $\mathbb{R}^n$. Moreover, let $\mathcal{X}_i \subset \mathcal{X}$ be the ith class containing N_i examples, with $\mathbf{x}_i^j$ denoting the jth example in $\mathcal{X}_i$. In this section, we briefly review two kernel subspace analysis methods, KPCA and KDA, performed on the data set $\mathcal{X}$.

2.1 KPCA Algorithm

Schölkopf *et al.* [24] first proposed KPCA as a nonlinear extension of PCA. We briefly review the method in this subsection.

Given the data set $\mathcal{X}$, the covariance matrix in $\mathcal{F}$, denoted $\mathbf{\Sigma}_\phi$, is given by

$$\begin{aligned}\mathbf{\Sigma}_\phi &= E\left[(\phi(\mathbf{x}) - E[\phi(\mathbf{x})])(\phi(\mathbf{x}) - E[\phi(\mathbf{x})])^T\right] \\ &= \frac{1}{N}\sum_{i=1}^{c}\sum_{j=1}^{N_i}(\phi(\mathbf{x}_i^j) - \mathbf{m}^\phi)(\phi(\mathbf{x}_i^j) - \mathbf{m}^\phi)^T, \end{aligned} \tag{1}$$

where $\mathbf{m}^\phi = \frac{1}{N}\sum_{i=1}^{c}\sum_{j=1}^{N_i}\phi(\mathbf{x}_i^j)$ is the mean over all N feature vectors in $\mathcal{F}$. Since KPCA seeks to find the optimal projection directions in $\mathcal{F}$, onto which all patterns are projected to give the corresponding covariance matrix with maximum trace, the objective function can be defined by maximizing the following:

$$J_{kpca}(\mathbf{v}) = \mathbf{v}^T\mathbf{\Sigma}_\phi\mathbf{v}. \tag{2}$$

Since the dimensionality of $\mathcal{F}$ is generally very high or even infinite, it is therefore inappropriate to solve the following eigenvalue problem for the solution as in traditional PCA:

$$\mathbf{\Sigma}_\phi\mathbf{v} = \lambda\mathbf{v}. \tag{3}$$

Fortunately, we can show that the eigenvector $\mathbf{v}$ must lie in a space spanned by $\{\phi(\mathbf{x}_i^j)\}$ in $\mathcal{F}$ and thus it can be expressed in the form of the following linear expansion:

$$\mathbf{v} = \sum_{i=1}^{c}\sum_{j=1}^{N_i} w_i^j\phi(\mathbf{x}_i^j). \tag{4}$$

Substituting (4) into (2), we obtain an equivalent eigenvalue problem as follows:

$$\left(\mathbf{I} - \frac{1}{N}\mathbf{1}\right)\mathbf{K}\left(\mathbf{I} - \frac{1}{N}\mathbf{1}\right)^T\mathbf{w} = \lambda\mathbf{w}, \tag{5}$$

where $\mathbf{I}$ is an $N \times N$ identity matrix, $\mathbf{1}$ is an $N \times N$ matrix with all terms being one, $\mathbf{w} = (w_1^1, \ldots, w_1^{N_1}, \ldots, w_c^1, \ldots, w_c^{N_c})^T$ is the vector of expansion coefficients of a given eigenvector $\mathbf{v}$, and $\mathbf{K}$ is the $N \times N$ Gram matrix which can be further defined as $\mathbf{K} = (\mathbf{K}_{lh})_{l,h=1,\ldots,c}$ where $\mathbf{K}_{lh} = (k_{ij})_{i=1,\ldots,N_l}^{j=1,\ldots,N_h}$ and $k_{ij} = \langle\phi(\mathbf{x}_l^i), \phi(\mathbf{x}_h^j)\rangle$.

The solution to (5) can be found by solving for the orthonormal eigenvectors $\mathbf{w}_1, \ldots, \mathbf{w}_m$ corresponding to the m largest eigenvalues $\lambda_1, \ldots, \lambda_m$, which are arranged in descending order. Thus, the eigenvectors of (3) can be obtained as $\mathbf{\Phi}\mathbf{w}_i \, (i = 1, \ldots, m)$, where $\mathbf{\Phi} = [\phi(\mathbf{x}_1^1), \ldots, \phi(\mathbf{x}_1^{N_1}), \ldots, \phi(\mathbf{x}_c^1), \ldots, \phi(\mathbf{x}_c^{N_c})]$. Furthermore, the corresponding normalized eigenvectors $\mathbf{v}_i \, (i = 1, \ldots, m)$ can be obtained as $\mathbf{v}_i = \frac{1}{\sqrt{\lambda_i}}\mathbf{\Phi}\mathbf{w}_i$, since $(\mathbf{\Phi}\mathbf{w}_i)^T\mathbf{\Phi}\mathbf{w}_i = \lambda_i$.

With $\mathbf{v}_i = \frac{1}{\sqrt{\lambda_i}}\mathbf{\Phi}\mathbf{w}_i \, (i = 1, \ldots, m)$ constituting the m orthonormal projection directions in $\mathcal{F}$, any novel input vector $\mathbf{x}$ can obtain its low-dimensional feature representation $\mathbf{y} = (y_1, \ldots, y_m)^T$ in $\mathcal{F}$ as:

$$\mathbf{y} = (\mathbf{v}_1, \ldots, \mathbf{v}_m)^T\phi(\mathbf{x}), \tag{6}$$

with each KPCA feature $y_i \, (i = 1, \ldots, m)$ expanded further as

$$\begin{aligned} y_i &= \mathbf{v}_i^T\phi(\mathbf{x}) = \frac{1}{\sqrt{\lambda_j}}\mathbf{w}_i^T\mathbf{\Phi}\phi(\mathbf{x}) \\ &= \frac{1}{\sqrt{\lambda_j}}\mathbf{w}_i^T(k(\mathbf{x}_1^1, \mathbf{x}), \ldots, k(\mathbf{x}_1^{N_1}, \mathbf{x}), \ldots, k(\mathbf{x}_c^1, \mathbf{x}), \ldots, k(\mathbf{x}_c^{N_c}, \mathbf{x})). \end{aligned} \tag{7}$$

Unlike traditional PCA which only captures second-order statistics in the input space, KPCA captures second-order statistics in the feature space which can correspond to higher-order statistics in the input space depending on the feature map (and hence kernel) used. Therefore, KPCA is superior to PCA in extracting more powerful features, which are especially essential when the face image variations due to illumination and pose are significantly complex and nonlinear. Nevertheless, KPCA is still an unsupervised learning method and hence the (nonlinear) features extracted by KPCA do not necessarily give rise to high separability between classes.

2.2 KDA Algorithm

Similar to the kernel extension of PCA to give KPCA, the kernel trick can also be applied to LDA to give its kernel extension. KDA seeks to find the optimal projection directions in $\mathcal{F}$ by simultaneously maximizing the between-class scatter and minimizing the within-class scatter in $\mathcal{F}$. In this subsection, we briefly review the KDA method.

For a data set $\mathcal{X}$ and a feature map ϕ, the between-class scatter matrix $\mathbf{S}_b^\phi$

and within-class scatter matrix $\mathbf{S}_w^\phi$ in $\mathcal{F}$ can be defined as:

$$\mathbf{S}_b^\phi = \sum_{i=1}^{c} \frac{N_i}{N} (\mathbf{m}_i^\phi - \mathbf{m}^\phi)(\mathbf{m}_i^\phi - \mathbf{m}^\phi)^T \tag{8}$$

and

$$\mathbf{S}_w^\phi = \frac{1}{N} \sum_{i=1}^{c} \sum_{j=1}^{N_i} (\phi(\mathbf{x}_i^j) - \mathbf{m}_i^\phi)(\phi(\mathbf{x}_i^j) - \mathbf{m}_i^\phi)^T, \tag{9}$$

where $\mathbf{m}_i^\phi = \frac{1}{N_i} \sum_{j=1}^{N_i} \phi(\mathbf{x}_i^j)$ is the class mean of $\mathcal{X}_i$ in $\mathcal{F}$ and $\mathbf{m}^\phi = \frac{1}{N} \sum_{i=1}^{c} \sum_{j=1}^{N_i} \phi(\mathbf{x}_i^j)$ is the overall mean as before.

Analogous to LDA which operates on the input space, the optimal projection directions for KDA can be obtained by maximizing the Fisher criterion in $\mathcal{F}$:

$$J(\mathbf{v}) = \frac{\mathbf{v}^T \mathbf{S}_b^\phi \mathbf{v}}{\mathbf{v}^T \mathbf{S}_w^\phi \mathbf{v}}. \tag{10}$$

Like KPCA, we do not solve this optimization problem directly due to the high or even infinite dimensionality of $\mathcal{F}$. As for KPCA, we can show that any solution $\mathbf{v} \in \mathcal{F}$ must lie in the space spanned by $\{\phi(\mathbf{x}_i^j)\}$ in $\mathcal{F}$ and thus it can be expressed as

$$\mathbf{v} = \sum_{i=1}^{c} \sum_{j=1}^{N_i} w_i^j \phi(\mathbf{x}_i^j). \tag{11}$$

Substituting (11) into the numerator and denominator of (10), we obtain

$$\mathbf{v}^T \mathbf{S}_b^\phi \mathbf{v} = \mathbf{w}^T \mathbf{K}_b \mathbf{w} \tag{12}$$

and

$$\mathbf{v}^T \mathbf{S}_w^\phi \mathbf{v} = \mathbf{w}^T \mathbf{K}_w \mathbf{w}, \tag{13}$$

where $\mathbf{w} = (w_1^1, \ldots, w_1^{N_1}, \ldots, w_c^1, \ldots, w_c^{N_C})^T$, and $\mathbf{K}_b$ and $\mathbf{K}_w$ can be seen as the variant scatter matrices based on some manipulation of the Gram matrix $\mathbf{K}$. As a result, the solution to (10) can be obtained by maximizing the following optimization problem instead:

$$J_f(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{K}_b \mathbf{w}}{\mathbf{w}^T \mathbf{K}_w \mathbf{w}}, \tag{14}$$

giving the m leading eigenvectors $\mathbf{w}_1, \ldots, \mathbf{w}_m$ of the matrix $\mathbf{K}_w^{-1} \mathbf{K}_b$ as solution.

For any input vector $\mathbf{x}$, its low-dimensional feature representation $\mathbf{y} = (y_1, \ldots, y_m)^T$ can then be obtained as

$$\mathbf{y} = (\mathbf{w}_1, \ldots, \mathbf{w}_m)^T (k(\mathbf{x}_1^1, \mathbf{x}), \ldots, k(\mathbf{x}_1^{N_1}, \mathbf{x}), \ldots, k(\mathbf{x}_c^1, \mathbf{x}), \ldots, k(\mathbf{x}_c^{N_c}, \mathbf{x}))^T. \tag{15}$$

Note that the solution above is based on the assumption that the within-class scatter matrix $\mathbf{K}_w$ is invertible. However, for face recognition applications,

this assumption is almost always invalid due to the undersampling problem as discussed above. One simple method for solving this problem is to use the pseudo-inverse of $\mathbf{K}_w$ instead. Another simple method is to add a small multiple of the identity matrix ($\epsilon\mathbf{I}$ for some small $\epsilon > 0$) to $\mathbf{K}_w$ to make it non-singular. Although more effective methods have been proposed (e.g., [26,19]), we keep it simple in this paper by adding $\epsilon\mathbf{I}$ to $\mathbf{K}_w$ where $\epsilon = 10^{-7}$ in our experiments.

3 Feature Extraction via Kernel Fractional-step Discriminant Analysis

3.1 Kernel Fractional-step Discriminant Analysis

The primary objective of the new KFDA algorithm is to overcome the adverse effects caused by outlier classes. A commonly adopted method to solve this problem is to incorporate a weighting function into the Fisher criterion by using a weighted between-class scatter matrix in place of the ordinary between-class scatter matrix, as in [7,8,10,15,23]. However, it is not clear how to set the weights in the weighting function appropriately to put more emphasis on those classes that are close together and hence are more likely to lead to misclassification. Our KFDA algorithm involves two steps. The first step is similar to the ordinary KDA procedure in obtaining a low-dimensional subspace and then the second step applies the FLDA procedure to further reduce the dimensionality by adjusting the weights in the weighting function automatically through making fractional steps.

As in [7,8,10], we define the weighted between-class scatter matrix in $\mathcal{F}$ as follows:

$$\mathbf{S}_B^\phi = \sum_{i=1}^{c-1} \sum_{j=i+1}^{c} \frac{N_i N_j}{N^2} w(d_{ij})(\mathbf{m}_i^\phi - \mathbf{m}_j^\phi)(\mathbf{m}_i^\phi - \mathbf{m}_j^\phi)^T, \tag{16}$$

where the weighting function $w(d_{ij})$ is a monotonically decreasing function of the Euclidean distance $d_{ij} = \|\mathbf{m}_i^\phi - \mathbf{m}_j^\phi\|$ with $\mathbf{m}_i^\phi$ and $\mathbf{m}_j^\phi$ being the class means for $\mathcal{X}_i$ and $\mathcal{X}_j$ in $\mathcal{F}$, respectively. Apparently, the weighted between-class scatter matrix $\mathbf{S}_B^\phi$ degenerates to the conventional between-class scatter matrix $\mathbf{S}_b^\phi$ if the weighting function in (16) always gives a constant weight value. In this sense $\mathbf{S}_B^\phi$ can be regarded as a generalization of $\mathbf{S}_b^\phi$. According to the FLDA procedure in [16], the weighting function should drop faster than the Euclidean distance between the class means for $\mathcal{X}_i$ and $\mathcal{X}_j$ in $\mathcal{F}$. As a result, the only constraint for the weighting function $w(d_{ij}) = d_{ij}^{-p}$, where $p \in \mathbb{N}$, is $p \geq 3$. Moreover, it is easy to note that d_{ij} in $\mathcal{F}$ can be computed by applying the kernel trick as follows:

$$
\begin{aligned}
d_{ij} &= \|\mathbf{m}_i^{\phi} - \mathbf{m}_j^{\phi}\| \\
&= \sqrt{\left(\sum_{\mathbf{x}_{i_1}\in\mathcal{X}_i} \frac{\phi(\mathbf{x}_{i_1})}{N_i} - \sum_{\mathbf{x}_{j_1}\in\mathcal{X}_j} \frac{\phi(\mathbf{x}_{j_1})}{N_j}\right)^T \left(\sum_{\mathbf{x}_{i_2}\in\mathcal{X}_i} \frac{\phi(\mathbf{x}_{i_2})}{N_i} - \sum_{\mathbf{x}_{j_2}\in\mathcal{X}_j} \frac{\phi(\mathbf{x}_{j_2})}{N_j}\right)} \\
&= \sqrt{\sum_{\mathbf{x}_{i_1},\mathbf{x}_{i_2}\in\mathcal{X}_i} \frac{k_{i_1,i_2}}{N_i^2} + \sum_{\mathbf{x}_{j_1},\mathbf{x}_{j_2}\in\mathcal{X}_j} \frac{k_{j_1,j_2}}{N_j^2} - \sum_{\mathbf{x}_{i_1}\in\mathcal{X}_i,\mathbf{x}_{j_2}\in\mathcal{X}_j} \frac{k_{i_1,j_2}}{N_iN_j} - \sum_{\mathbf{x}_{i_2}\in\mathcal{X}_i,\mathbf{x}_{j_1}\in\mathcal{X}_j} \frac{k_{j_1,i_2}}{N_iN_j}},
\end{aligned} \tag{17}
$$

where $k_{i_1,i_2} = k(\mathbf{x}_i^{i_1}, \mathbf{x}_i^{i_2}) = \langle\phi(\mathbf{x}_i^{i_1}), \phi(\mathbf{x}_i^{i_2})\rangle$, $k_{j_1,j_2} = k(\mathbf{x}_j^{j_1}, \mathbf{x}_j^{j_2}) = \langle\phi(\mathbf{x}_j^{j_1}), \phi(\mathbf{x}_j^{j_2})\rangle$, $k_{i_1,j_2} = k(\mathbf{x}_i^{i_1}, \mathbf{x}_j^{j_2}) = \langle\phi(\mathbf{x}_i^{i_1}), \phi(\mathbf{x}_j^{j_2})\rangle$, and $k_{j_1,i_2} = k(\mathbf{x}_j^{j_1}, \mathbf{x}_i^{i_2}) = \langle\phi(\mathbf{x}_j^{j_1}), \phi(\mathbf{x}_i^{i_2})\rangle$.

Based on the definition of $\mathbf{S}_B^{\phi}$ in (16), we define a new Fisher criterion in $\mathcal{F}$ as

$$
\hat{J}(\mathbf{v}) = \frac{\mathbf{v}^T\mathbf{S}_B^{\phi}\mathbf{v}}{\mathbf{v}^T\mathbf{S}_w^{\phi}\mathbf{v}}. \tag{18}
$$

Again, we can express the solution $\mathbf{v} = \sum_{i=1}^{c}\sum_{j=1}^{N_i} w_i^j\phi(\mathbf{x}_i^j)$ and hence rewrite the Fisher criterion in (18) as (see Appendix A.1) [1]

$$
\hat{J}_f(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{K}_B\mathbf{w}}{\mathbf{w}^T\mathbf{K}_w\mathbf{w}}, \tag{19}
$$

where

$$
\mathbf{w} = (w_1^1, \ldots, w_1^{N_1}, \ldots, w_c^1, \ldots, w_c^{N_c})^T, \tag{20}
$$

$$
\mathbf{K}_B = \sum_{i=1}^{c-1}\sum_{j=i+1}^{c} \frac{N_iN_j}{N^2} w(d_{ij})(\mathbf{m}_i - \mathbf{m}_j)(\mathbf{m}_i - \mathbf{m}_j)^T, \tag{21}
$$

$$
\mathbf{K}_w = \frac{1}{N}\sum_{i=1}^{c}\sum_{j=1}^{N_i}(\mathbf{k}_i^j - \mathbf{m}_i)(\mathbf{k}_i^j - \mathbf{m}_i)^T, \tag{22}
$$

with

[1] Since the term $\mathbf{K}_w$ is the same as that in classical KDA, Appendix A.1 only provides the derivation for $\mathbf{K}_B$ in (21).

$$\mathbf{m}_i = \Bigg(\frac{1}{N_i} \sum_{h=1}^{N_i} k(\mathbf{x}_1^1, \mathbf{x}_i^h), \ldots, \frac{1}{N_i} \sum_{h=1}^{N_i} k(\mathbf{x}_1^{N_1}, \mathbf{x}_i^h), \ldots, \frac{1}{N_i} \sum_{h=1}^{N_i} k(\mathbf{x}_c^1, \mathbf{x}_i^h), \ldots, \frac{1}{N_i} \sum_{h=1}^{N_i} k(\mathbf{x}_c^{N_c}, \mathbf{x}_i^h) \Bigg)^T, \tag{23}$$

$$\mathbf{m}_j = \Bigg(\frac{1}{N_j} \sum_{h=1}^{N_j} k(\mathbf{x}_1^1, \mathbf{x}_j^h), \ldots, \frac{1}{N_j} \sum_{h=1}^{N_j} k(\mathbf{x}_1^{N_1}, \mathbf{x}_j^h), \ldots, \frac{1}{N_j} \sum_{h=1}^{N_j} k(\mathbf{x}_c^1, \mathbf{x}_j^h), \ldots, \frac{1}{N_j} \sum_{h=1}^{N_j} k(\mathbf{x}_c^{N_c}, \mathbf{x}_j^h) \Bigg)^T, \tag{24}$$

$$\mathbf{k}_i^j = (k(\mathbf{x}_1^1, \mathbf{x}_i^j), \ldots, k(\mathbf{x}_1^{N_1}, \mathbf{x}_i^j), \ldots, k(\mathbf{x}_c^1, \mathbf{x}_i^j), \ldots, k(\mathbf{x}_c^{N_c}, \mathbf{x}_i^j))^T. \tag{25}$$

The solution to (18) is thus the m leading eigenvectors $\mathbf{w}_1, \ldots, \mathbf{w}_m$ of the matrix $\mathbf{K}_w^{-1}\mathbf{K}_B$.

For any input vector $\mathbf{x}$, its low-dimensional feature representation $\mathbf{z} = (z_1, \ldots, z_m)^T$ can thus be given by (see Appendix A.2)

$$\mathbf{z} = (\mathbf{w}_1, \ldots, \mathbf{w}_m)^T (k(\mathbf{x}_1^1, \mathbf{x}), \ldots, k(\mathbf{x}_1^{N_1}, \mathbf{x}), \ldots, k(\mathbf{x}_c^1, \mathbf{x}), \ldots, k(\mathbf{x}_c^{N_c}, \mathbf{x}))^T. \tag{26}$$

Through the weighted KDA procedure described above, we obtain a low-dimensional subspace where almost all classes become linearly separable, although some classes remain closer to each other than others. In what follows, an FLDA step is directly applied to further reduce the dimensionality of this subspace from m to the required m' through making fractional steps. The primary motivation for FLDA comes from the following consideration [16]. In order to substantially improve the robustness of the choice of the weighting function and avoid the instability of the algorithm brought by an inaccurate weighting function [10], FLDA introduces some sort of automatic gain control, which reduces the dimensionality in small fractional steps rather than integral steps. This allows the between-class scatter matrix and its eigenvectors to be iteratively recomputed in accordance with the variations of the weighting function, so that the chance of overlap between classes can be reduced. Therefore, in the output classification space, FLDA can increase the separability between classes that have small inter-class distances in the input space while preserve the high separability between classes that are already far apart. Furthermore, unlike some other techniques [10,15,23], FLDA computes the weighting function without having to adopt any restrictive statistical model assumption, making it applicable to more general problem settings including face recognition tasks with high variability between images.

Figure A.1 summarizes the KFDA algorithm for feature extraction which is

used with the nearest neighbor rule for classification. In the next section, we propose a further extension of the KFDA algorithm by introducing two new kernel functions.

3.2 New Kernel Functions

Kernel subspace analysis methods make use of the input data for the computation exclusively in the form of inner products in $\mathcal{F}$, with the inner products computed implicitly via a kernel function, called Mercer kernel, $k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$, where $\langle \cdot, \cdot \rangle$ is an inner product operator in $\mathcal{F}$. A symmetric function is a Mercer kernel if and only if the Gram matrix formed by applying this function to any finite subset of $\mathcal{X}$ is positive semi-definite. Some kernel functions such as the polynomial kernel, Gaussian RBF kernel and sigmoid kernel have been commonly used in many practical applications of kernel methods. For face recognition using kernel subspace analysis methods in particular [6–8,13,17,29,31,32], the polynomial kernel and Gaussian RBF kernel have been used extensively to demonstrate that kernel subspace analysis methods are more effective than their linear counterparts in many cases. While in principle any Mercer kernel can be used with kernel subspace analysis methods, not all kernel functions are equally good in terms of the performance that the kernel methods can deliver.

Recently, the feasibility and effectiveness of different kernel choices for kernel subspace analysis methods has been investigated in the context of face recognition applications. Chen *et al.* [5] proposed a KDA-based method for face recognition with the parameter of the Gaussian RBF kernel determined in advance. Yang *et al.* [26] proposed using a kernel function for KDA by combining multiple Gaussian RBF kernels with different parameters, with the combination coefficients determined to optimally combine the individual kernels. However, the effectiveness of combining multiple kernels is still implicit for KDA, since the conventional Fisher criterion in $\mathcal{F}$ is converted into the two-dimensional Fisher criterion at the same time. Liu *et al.* [14] extended the polynomial kernel to the so-called cosine polynomial kernel by applying the cosine measure. One motivation is that the inner product of two vectors in $\mathcal{F}$ can be regarded as a similarity measure between them. Another motivation is that such measure has been found to perform well in practice. Moreover, Liu [11] extended the ordinary polynomial kernel in an innovative way to include the fractional-power polynomial kernel as well. However, the fractional-power polynomial kernel proposed, like the sigmoid kernel, is not a Mercer kernel. Nevertheless, Liu applied KPCA using the fractional-power polynomial kernel for face recognition and showed improvement in performance when compared with the ordinary polynomial kernel.

Inspired by the effectiveness of the cosine measure for kernel functions [14], we further extend the fractional-power polynomial kernel in this paper by incorporating the cosine measure to give the so-called cosine fractional-power polynomial kernel. Note that the cosine measure can only be used with non-stationary kernels such as the polynomial kernel, but not with isotropic kernels such as the Gaussian RBF kernel that only depend on the difference between two vectors. To improve the face recognition performance of the Gaussian RBF kernel, we also go beyond the traditional Gaussian RBF kernel by considering non-normal Gaussian RBF kernels, $k_{RBF}(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^d/\sigma^2)$ where $d \geq 0$ and $d \neq 2$. According to [28], non-normal Gaussian RBF kernels completely satisfy the Mercer condition and hence are Mercer kernels if and only if $0 \leq d \leq 2$.

In summary, the kernel functions considered in this paper are listed below:

- Polynomial kernel (poly):

$$k_{poly}(\mathbf{x}, \mathbf{y}) = (\gamma_1 \mathbf{x} \cdot \mathbf{y}^T - \gamma_2)^d \quad \text{with } d \in \mathbb{N} \text{ and } d \geq 1. \tag{27}$$

- Fractional-power polynomial kernel (fp-poly):

$$k_{fp-poly}(\mathbf{x}, \mathbf{y}) = (\gamma_1 \mathbf{x} \cdot \mathbf{y}^T - \gamma_2)^d \quad \text{with } 0 < d < 1. \tag{28}$$

- Cosine polynomial kernel (c-poly):

$$k_{c-poly}(\mathbf{x}, \mathbf{y}) = \frac{k_{poly}(\mathbf{x}, \mathbf{y})}{\sqrt{k_{poly}(\mathbf{x}, \mathbf{x}) k_{poly}(\mathbf{y}, \mathbf{y})}}. \tag{29}$$

- Cosine fractional-power polynomial kernel (cfp-poly):

$$k_{cfp-poly}(\mathbf{x}, \mathbf{y}) = \frac{k_{fp-poly}(\mathbf{x}, \mathbf{y})}{\sqrt{k_{fp-poly}(\mathbf{x}, \mathbf{x}) k_{fp-poly}(\mathbf{y}, \mathbf{y})}}. \tag{30}$$

- Gaussian RBF kernel (RBF):

$$k_{RBF}(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^2/\sigma^2). \tag{31}$$

- Non-normal Gaussian RBF kernel (NN-RBF):

$$k_{nn-RBF}(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^d/\sigma^2) \quad \text{with } d \geq 0 \text{ and } d \neq 2. \tag{32}$$

4 Experimental Results

4.1 Visualization of Data Distributions

We first compare the distribution of data points after applying FLDA, KDA and KFDA, respectively. For visualization sake, we reduce the dimensionality to 2. Two databases are used for this set of experiments. They are the image segmentation database from the UCI Machine Learning Repository [2] and the YaleB face database. [3]

The instances in the image segmentation database are drawn randomly from a database of seven outdoor images. Each instance is represented by 19 continuous-valued features extracted from a 3×3 region, and the corresponding class label is obtained by manual segmentation into seven classes: brick-face, sky, foliage, cement, window, path and grass. The database consists of 30 instances for each of the seven classes, summing up to a total of 210 instances for the whole database. Since the dimensionality of the instances in the image segmentation database is relatively low, FLDA can be applied to the instances directly. We also follow the suggestion of [16] to set the number of fractional steps to 30 in our experiments. Figure A.2(a)-(c) depict the distributions of data points based on the two most discriminant features after applying FLDA, KDA and KFDA, respectively. For KDA and KFDA, the Gaussian RBF kernel $k(\mathbf{x}_1, \mathbf{x}_2) = \exp(-||\mathbf{x}_1 - \mathbf{x}_2||^2/10^4)$ is used. Moreover, following [16], the weighting functions in both FLDA and KFDA are set to $w(d_{ij}) = d_{ij}^{-12}$. We can see from Figure A.2(a) that most classes still remain nonseparable even after applying FLDA. For KDA, Figure A.2(b) shows that many classes become more linearly separable. However, some classes (such as brick-face, foliage, cement and window) are so closely clustered together that they cannot be perfectly separated. On the other hand, Figure A.2(c) shows that all classes are very well separated and are more equally spaced when KFDA is applied.

We further do some experiments on the YaleB face database [9]. The YaleB face database contains 5850 face images of 10 subjects each captured under 585 different viewing conditions (9 poses $\times$ 65 illumination conditions). To facilitate visualization, we select a subset of 240 face images from the database with 30 images for each of eight humans. All images have been manually cropped and then normalized to a size of 46×56 with 256 gray levels. Figure A.3 shows the 30 images of one individual used in the experiments. Since the face patterns are high-dimensional, we use DFLDA [18] in place of FLDA [16] for this set of experiments. Figure A.4(a)-(c) depict the distributions of data points based on the two most discriminant features after applying DFLDA,

[2] http://www.ics.uci.edu/∼mlearn/MLRepository
[3] http://cvc.yale.edu/projects/yalefacesB/yalefacesB.html

KDA and KFDA, respectively. For KDA and KFDA, the Gaussian RBF kernel $k(\mathbf{x}_1, \mathbf{x}_2) = \exp(-||\mathbf{x}_1 - \mathbf{x}_2||^2/10^9)$ is used. As above, the weighting functions in both DFLDA and KFDA are set to $w(d_{ij}) = d_{ij}^{-12}$. The findings are similar to those for the image segmentation data above. While KDA gives classes that are more compact than those obtained by DFLDA, the classes obtained by KFDA are both more compact and well separated from each other, as shown in Figure A.4(c).

4.2 Face Recognition Experiments

In this subsection, we study the face recognition performance of KFDA with the different kernel functions given in Section 3.2 and compare it with two other kernel methods, KPCA and KDA. We use the FERET face database for these experiments. [4]

The FacE REcognition Technology (FERET) face database [22] is from the FERET Program sponsored by the US Department of Defense's Counterdrug Technology Development Program through the Defense Advanced Research Projects Agency (DARPA), and it has become the de facto standard for evaluating state-of-the-art face recognition algorithms. The whole database contains 13,539 face images of 1,565 subjects taken during different photo sessions with variations in size, pose, illumination, facial expression, and even age. The subset we use in our experiments includes 200 subjects each with four different images. All images are obtained by cropping based on the manually located centers of the eyes, and are normalized to the same size of 92×112 with 256 gray levels. Figure A.5 shows some sample images used in our experiments.

In the following experiments, the images for each subject are randomly partitioned into two disjoint sets for training and testing. More specifically, three images per subject are randomly chosen from the four images available for each subject for training while the rest for testing. For each feature representation obtained by a dimensionality reduction method, we use the nearest neighbor rule with Euclidean distance measure to assess the classification accuracy. Each experiment is repeated 10 times and the average classification rates are reported. We set $\gamma_1 = 10^{-9}$ and $\gamma_2 = 1$ for the first four non-stationary kernels presented in Section 3.2 and $\sigma^2 = 10^9$ for the last two isotropic kernels.

We first compare KFDA with KDA using different non-stationary kernels: polynomial kernel (poly), fractional-power polynomial kernel (fp-poly), cosine polynomial kernel (c-poly), and cosine fractional-power polynomial kernel (cfp-poly). As in [16], we set the weighting function for KFDA as $w(d_{ij}) = d_{ij}^{-12}$. The only remaining parameter in the different non-stationary kernels is the

[4] http://www.itl.nist.gov/iad/humanid/feret/

exponent d. For simplicity, we only study the polynomial (poly) and cosine polynomial (c-poly) kernels with exponents $d = \{1, 2\}$ and the fractional-power polynomial (fp-poly) and cosine fractional-power polynomial (cfp-poly) kernels with exponents $d = \{0.8, 0.6, 0.4\}$. We have also tested the polynomial and cosine polynomial kernels with $d = \{3, 4, 5\}$ but the results are similar to those for $d = 2$. Hence, we do not include their results in the comparison. Figure A.6 compares KDA and KFDA on different non-stationary kernels. We have also tried KPCA but its results are very poor compared with KDA and KFDA and hence are not included in the graphs for comparison. Figure A.6(a)-(c) show the performance of KDA on the four different non-stationary kernels. The fractional-power polynomial kernel outperforms the polynomial kernel and the cosine fractional-power polynomial kernel can achieve further improvement. The performance of KFDA on different non-stationary kernels shows a similar trend, as we can see in Figure A.6(d)-(f). Finally, Figure A.6(g) and (h) show that KFDA outperforms KDA because KFDA can resist the adverse effects due to outlier classes.

We next compare KDA and KFDA on different isotropic kernels, which include the Gaussian RBF kernel (RBF) and the non-normal Gaussian RBF kernel (NN-RBF). Figure A.7 shows the results. The weighting function for KFDA is still set to $w(d_{ij}) = d_{ij}^{-12}$ as before and the non-normal Gaussian RBF kernel is tested on exponents $d = \{1.5, 1.6, 1.8\}$. Figure A.7(a) and (b) show that the non-normal Gaussian RBF kernel generally gives better results than the Gaussian RBF kernel for both KDA and KFDA. In Figure A.7(c), we can see that KFDA significantly outperforms KDA when the Gaussian RBF kernel is used. This is also true for the non-normal Gaussian RBF kernel, as shown in Figure A.7(d). This shows that KFDA can successfully resist the adverse effects due to outlier classes and hence can improve the face recognition accuracy significantly. We have also performed some experiments for both KDA and KFDA on the non-normal Gaussian RBF kernel with $d > 2$. Except for a few cases, the recognition rates are generally far lower than those for the Gaussian RBF kernel. A possible explanation is that the kernel is no longer a Mercer kernel when $d > 2$.

We further try KFDA on some other weighting functions $w(d_{ij}) = \{d_{ij}^{-4}, d_{ij}^{-6}, d_{ij}^{-8}, d_{ij}^{-10}\}$ as recommended by [16]. Table A.1 shows the results comparing KFDA on different weighting functions $w(d_{ij}) = \{d_{ij}^{-4}, d_{ij}^{-8}, d_{ij}^{-12}\}$ with KPCA and KDA. We can see that KFDA is superior to KPCA and KDA on different kernels for all weighting functions tried. A closer look shows that the cosine fractional-power polynomial kernel and the non-normal Gaussian RBF kernel can effectively boost the performance of KFDA. Specifically, KFDA with the cosine fractional-power polynomial kernel for $d = 0.4$ and the weighting function $w(d_{ij}) = d_{ij}^{-12}$ achieves a recognition rate of 90% using 62 features. Using the non-normal Gaussian RBF kernel with $d = 1.8$ and the weighting function $w(d_{ij}) = d_{ij}^{-12}$, it gives a recognition rate of 91.65% using only 46 features. On

the other hand, KPCA requires a lot more features for all kernels and yet the results are not satisfactory. Except for the non-normal Gaussian RBF kernel, KDA using other kernels also requires many more features than KFDA to attain the highest recognition rates and yet it is still inferior to KFDA.

To sufficiently evaluate the overall performance of KFDA, we follow the suggestions of [17,18] to also report the average percentages of the error rate of KFDA over those of both KPCA and KDA on different kernels. Under our experimental settings, the average percentage of the error rate of KFDA over that of another method can be calculated as the average of $(100-\alpha_i)/(100-\beta_i)$ $(i = 5, \ldots, 198)$, where α_i and β_i are the recognition rates in percentage of KFDA and another method, respectively, when i features are used. The corresponding results are summarized in Table A.2, where, as recommended by [16], the weighting functions $w(d_{ij}) = \{d_{ij}^{-4}, d_{ij}^{-8}, d_{ij}^{-12}\}$ are used for KFDA. It is clear from the results that the weighting scheme in KFDA can bring about performance improvement over KDA.

5 Conclusion

In this paper, we have proposed a novel kernel-based feature extraction method called KFDA. Not only can this new method deal with nonlinearity in a disciplined manner that is computationally attractive, it can also outperform traditional KDA algorithms in resisting the adverse effects due to outlier classes by incorporating a weighted between-class scatter matrix and adjusting its weights via making fractional steps in dimensionality reduction. We then further improve the performance of KFDA by using two new kernel functions: cosine fractional-power polynomial kernel and non-normal Gaussian RBF kernel. Extensive face recognition experiments based on the YaleB and FERET face databases show that KFDA significantly outperforms KPCA and is also superior to KDA for all the kernel functions tried. Moreover, the new kernels outperform the polynomial kernel and Gaussian RBF kernel.

Acknowledgments

The research described in this paper has been supported by Competitive Earmarked Research Grant (CERG) HKUST621305 from the Research Grants Council (RGC) of the Hong Kong Special Administrative Region, China.

References

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

[2] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman. Eigenfaces vs. Fisherfaces: recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:711–720, July 1997.

[3] H. Cevikalp, M. Neamtu, M. Wilkes, and A. Barkana. Discriminative common vectors for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(1):4–13, January 2005.

[4] L.F. Chen, H.Y.M. Liao, M.T. Ko, J.C. Lin, and G.J. Yu. A new LDA-based face recognition system which can solve the small sample size problem. *Pattern Recognition*, 33(10):1713–1726, 2000.

[5] W.S. Chen, P.C. Yuen, J. Huang, and D.Q. Dai. Kernel machine-based one-parameter regularized fisher discriminant method for face recognition. *IEEE Transactions on System, Man, and Cybernetics-Part B: Cybernetics*, 35(4):659–669, August 2005.

[6] G. Dai and Y.T. Qian. Kernel generalized nonlinear discriminant analysis algorithm for pattern recognition. In *Proceedings of the IEEE International Conference on Image Processing*, pages 2697–2700, 2004.

[7] G. Dai, Y.T. Qian, and S.Jia. A kernel fractional-step nonlinear discriminant analysis for pattern recognition. In *Proceedings of the Eighteenth International Conference on Pattern Recognition*, volume 2, pages 431–434, August 2004.

[8] G. Dai and D.Y. Yeung. Nonlinear dimensionality reduction for classification using kernel weighted subspace method. In *Proceedings of the IEEE International Conference on Image Processing*, pages 838–841, September 2005.

[9] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.

[10] Y.X. Li, Y.Q. Gao, and H. Erdogan. Weighted pairwise scatter to improve linear discriminant analysis. In *Proceedings of the 6th International Conference on Spoken Language Processing*, 2000.

[11] C.J. Liu. Gabor-based kernel PCA with fractional power polynomial models for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):572–581, May 2004.

[12] C.j. Liu and H. Wechsler. Robust coding schemes for indexing and retrieval from large face databases. *IEEE Transactions on Image Processing*, 9(1):132–137, January 2000.

[13] Q.S. Liu, R. Huang, H.Q. Lu, and S.D. Ma. Face recognition using kernel based fisher discriminant analysis. In *Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, pages 187–191, May 2002.

[14] Q.S. Liu, H.Q. Lu, and S.D. Ma. Improving kernel Fisher discriminant analysis for face recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1):42–49, January 2004.

[15] M. Loog, R.P.W. Duin, and R. Haeb-Umbach. Multiclass linear dimension reduction by weighted pairwise Fisher criteria. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(7):762–766, July 2001.

[16] R. Lotlikar and R. Kothari. Fractional-step dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(6):623–627, June 2000.

[17] J.W. Lu, K.N. Plataniotis, and A.N. Venetsanopoulos. Face recognition using kernel direct discriminant analysis algorithms. *IEEE Transactions on Neural Networks*, 14(1):117–126, January 2003.

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

[19] J.W. Lu, K.N. Plataniotis, A.N. Venetsanopoulos, and J. Wang. An efficient kernel discriminant analysis method. *Pattern Recognition*, 38(10):1788–1790, October 2005.

[20] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.R. Müller. Fisher discriminant analysis with kernels. In Y.H. Hu, J. Larsen, E. Wilson, and S. Douglas, editors, *Proceedings of the Neural Networks for Signal Processing IX*, pages 41–48, 1999.

[21] B. Moghaddam. Principal manifolds and probabilistic subspaces for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(6):780–788, June 2002.

[22] A.P.J. Phillips, H.Moon, P.J. Rauss, and S. Rizvi. The FERET evaluation methodology for face recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1090–1104, October 2000.

[23] A.K. Qin, P.N. Suganthan, and M. Loog. Uncorrelated heteroscedastic LDA based on the weighted pairwise Chernoff criterion. *Pattern Recognition*, 38(4):613–616, 2005.

[24] B. Schölkopf, A. Smola, and K.R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1999.

[25] D.L. Swets and J.J. Weng. Using discriminant eigenfeatures for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):831–836, August 1996.

[26] S.Yang, S.C Yan, D. Xu, X.O. Tang, and C. Zhang. Fisher+kernel criterion for discriminant analysis. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 197–202, June 2005.

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

[28] V. Vapnik. *Statistical learning theory*. Wiley, New York, 1998.

[29] J. Yang, A.F. Frangi, J.Y. Yang, D. Zhang, and Z. Jin. KPCA plus LDA: a complete kernel Fisher discriminant framework for feature extraction and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(2):230–244, February 2005.

[30] J. Yang and J.Y. Yang. Why can LDA be performed in PCA transformed space? *Pattern Recognition*, 36(2):563–566, 2003.

[31] M.H. Yang. Kernel eigenfaces vs. kernel Fisherfaces: face recognition using kernel methods. In *Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, pages 215–220, May 2002.

[32] M.H. Yang, N. Ahuja, and D. Kriegman. Face recognition using kernel eigenfaces. In *Proceedings of the IEEE International Conference on Image Processing*, pages 37–40, 2000.

[33] J. Ye, R. Janardan, C.H. Park, and H. Park. An optimization criterion for generalized discriminant analysis on undersampled problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8):982–994, August 2004.

[34] J. Ye and Q. Li. A two-stage linear discriminant analysis via QR-decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):929–941, June 2005.

[35] H. Yu and J. Yang. A direct LDA algorithm for high-dimensional data with application to face recognition. *Pattern Recognition*, 34(11):2067–2070, 2001.

A Detailed Derivations

A.1 Derivation of Equation (21)

For clarity, let us denote $\mathbf{\Phi} = (\phi(\mathbf{x}_1^1), \ldots, \phi(\mathbf{x}_1^{N_1}), \ldots, \phi(\mathbf{x}_c^1), \ldots, \phi(\mathbf{x}_c^{N_c}))$. Then we have $\mathbf{v} = \mathbf{\Phi}\mathbf{w}$.

Based on the relationship between (18) and (19), we have $\mathbf{K}_B = \mathbf{\Phi}^T \mathbf{S}_B^\phi \mathbf{\Phi}$.

To derive (21), $\mathbf{K}_B$ above can be expanded as

$$\begin{aligned}\mathbf{K}_B =& \mathbf{\Phi}^T \left(\sum_{i=1}^{c-1} \sum_{j=i+1}^{c} \frac{N_i N_j}{N^2} w(d_{ij}) (\mathbf{m}_i^\phi - \mathbf{m}_j^\phi)(\mathbf{m}_i^\phi - \mathbf{m}_j^\phi)^T \right) \mathbf{\Phi} \\ =& \sum_{i=1}^{c-1} \sum_{j=i+1}^{c} \frac{N_i N_j}{N^2} w(d_{ij}) (\mathbf{\Phi}^T \mathbf{m}_i^\phi - \mathbf{\Phi}^T \mathbf{m}_j^\phi)(\mathbf{\Phi}^T \mathbf{m}_i^\phi - \mathbf{\Phi}^T \mathbf{m}_j^\phi)^T.\end{aligned}$$

Since $\mathbf{m}_i^\phi = \frac{1}{N_i} \sum_{h=1}^{N_i} \phi(\mathbf{x}_i^h)$, the term $\mathbf{\Phi}^T \mathbf{m}_i^\phi$ can be expanded as

$$\begin{aligned}\mathbf{\Phi}^T \mathbf{m}_i^\phi =& \left(\mathbf{m}_i^{\phi^T} \phi(\mathbf{x}_1^1), \ldots, \mathbf{m}_i^{\phi^T} \phi(\mathbf{x}_1^{N_1}), \ldots, \mathbf{m}_i^{\phi^T} \phi(\mathbf{x}_c^1), \ldots, \mathbf{m}_i^{\phi^T} \phi(\mathbf{x}_c^{N_c}) \right)^T \\ =& \left(\frac{1}{N_i} \sum_{h=1}^{N_i} \phi(\mathbf{x}_i^h)^T \phi(\mathbf{x}_1^1), \ldots, \frac{1}{N_i} \sum_{h=1}^{N_i} \phi(\mathbf{x}_i^h)^T \phi(\mathbf{x}_1^{N_1}), \ldots, \right. \\ & \left. \frac{1}{N_i} \sum_{h=1}^{N_i} \phi(\mathbf{x}_i^h)^T \phi(\mathbf{x}_c^1), \ldots, \frac{1}{N_i} \sum_{h=1}^{N_i} \phi(\mathbf{x}_i^h)^T \phi(\mathbf{x}_c^{N_c}) \right)^T \\ =& \left(\frac{1}{N_i} \sum_{h=1}^{N_i} k(\mathbf{x}_1^1, \mathbf{x}_i^h), \ldots, \frac{1}{N_i} \sum_{h=1}^{N_i} k(\mathbf{x}_1^{N_1}, \mathbf{x}_i^h), \ldots, \right. \\ & \left. \frac{1}{N_i} \sum_{h=1}^{N_i} k(\mathbf{x}_c^1, \mathbf{x}_i^h), \ldots, \frac{1}{N_i} \sum_{h=1}^{N_i} k(\mathbf{x}_c^{N_c}, \mathbf{x}_i^h) \right)^T.\end{aligned}$$

Similarly, the term $\mathbf{\Phi}^T \mathbf{m}_j^\phi$ can be expanded as

$$\mathbf{\Phi}^T\mathbf{m}_j^{\phi} = \left(\frac{1}{N_j}\sum_{h=1}^{N_j} k(\mathbf{x}_1^1, \mathbf{x}_j^h), \ldots, \frac{1}{N_j}\sum_{h=1}^{N_j} k(\mathbf{x}_1^{N_1}, \mathbf{x}_j^h), \ldots, \right.$$
$$\left. \frac{1}{N_j}\sum_{h=1}^{N_j} k(\mathbf{x}_c^1, \mathbf{x}_j^h), \ldots, \frac{1}{N_j}\sum_{h=1}^{N_j} k(\mathbf{x}_c^{N_c}, \mathbf{x}_j^h) \right)^T .$$

For clarity, we denote the expansion of $\mathbf{\Phi}^T\mathbf{m}_i^{\phi}$ by $\mathbf{m}_i$ and the expansion of $\mathbf{\Phi}^T\mathbf{m}_j^{\phi}$ by $\mathbf{m}_j$.

Thus,

$$\mathbf{K}_B = \sum_{i=1}^{c-1}\sum_{j=i+1}^{c} \frac{N_i N_j}{N^2} w(d_{ij})(\mathbf{m}_i - \mathbf{m}_j)(\mathbf{m}_i - \mathbf{m}_j)^T.$$

Hence we can obtain equation (21).

A.2 Derivation of Equation (26)

Let $\mathbf{\Phi} = (\phi(\mathbf{x}_1^1), \ldots, \phi(\mathbf{x}_1^{N_1}), \ldots, \phi(\mathbf{x}_c^1), \ldots, \phi(\mathbf{x}_c^{N_c}))$ and $\mathbf{v}_i = \mathbf{\Phi}\mathbf{w}_i \, (i = 1, \ldots, m)$.

Thus, $\mathbf{v}_i = \mathbf{\Phi}\mathbf{w}_i \, (i = 1, \ldots, m)$ constitute the discriminant vectors with respect to the Fisher criterion $\hat{J}(\mathbf{v})$ in (18).

Then, the low-dimensional feature representation $\mathbf{z} = (z_1, \ldots, z_m)^T$ of the vector $\mathbf{x}$ can be calculated as

$$\begin{aligned}
\mathbf{z} &= (\mathbf{v}_1, \ldots, \mathbf{v}_m)^T\phi(\mathbf{x}) \\
&= (\mathbf{w}_1, \ldots, \mathbf{w}_m)^T\mathbf{\Phi}^T\phi(\mathbf{x}) \\
&= (\mathbf{w}_1, \ldots, \mathbf{w}_m)^T(\phi(\mathbf{x})^T\phi(\mathbf{x}_1^1), \ldots, \phi(\mathbf{x})^T\phi(\mathbf{x}_1^{N_1}), \ldots, \phi(\mathbf{x})^T\phi(\mathbf{x}_c^1), \ldots, \phi(\mathbf{x})^T\phi(\mathbf{x}_c^{N_c}))^T \\
&= (\mathbf{w}_1, \ldots, \mathbf{w}_m)^T(k(\mathbf{x}_1^1, \mathbf{x}), \ldots, k(\mathbf{x}_1^{N_1}, \mathbf{x}), \ldots, k(\mathbf{x}_c^1, \mathbf{x}), \ldots, k(\mathbf{x}_c^{N_c}, \mathbf{x}))^T.
\end{aligned}$$

Equation (26) can thus be obtained.

Training

Input: A set of training images $\mathcal{X} = \{\mathbf{x}_i^j \,|\, \mathbf{x}_i^j \in \mathbb{R}^n, i = 1, \ldots, c, j = 1, \ldots, N_i\}$.

Output: A set of low-dimensional feature vectors $\mathcal{Y} = \{\mathbf{y}_i^j \,|\, \mathbf{y}_i^j \in \mathbb{R}^{m'}, i = 1, \ldots, c, j = 1, \ldots, N_i\}$ with enhanced discrimination power.

- Calculate the weighting function value $w(d_{ij})$ for all pairs of classes by (17).
- Calculate the matrices $\mathbf{K}_B$ and $\mathbf{K}_w$ by (21) and (22), respectively.
- Project all training examples in $\mathcal{X}$ to a lower-dimensional subspace by (26) to obtain $\{\mathbf{z}_i^j \in \mathbb{R}^m\}$.
- Apply FLDA to further reduce the dimensionality of $\{\mathbf{z}_i^j \in \mathbb{R}^m\}$ from m to m'. Let $\mathbf{\Xi}$ denote the transformation matrix to transform the m-dimensional vectors to m'-dimensional vectors in the final output subspace.
- Apply $\mathbf{\Xi}$ to each $\mathbf{z}_i^j$ as $\mathbf{y}_i^j = \mathbf{\Xi}\mathbf{z}_i^j$ to obtain the set of low-dimensional feature vectors $\mathcal{Y}$.

Classification

- Given any test example $\mathbf{x}$, its low-dimensional feature representation $\mathbf{y}$ can be computed as $\mathbf{y} = \mathbf{\Xi}\mathbf{z}$, with $\mathbf{z}$ obtained by applying (26) to $\mathbf{x}$.
- Apply the nearest neighbor rule to assign $\mathbf{y}$ to the same class as the nearest neighbor of $\mathbf{y}$ in $\mathcal{Y}$.

Fig. A.1. Summary of KFDA algorithm.

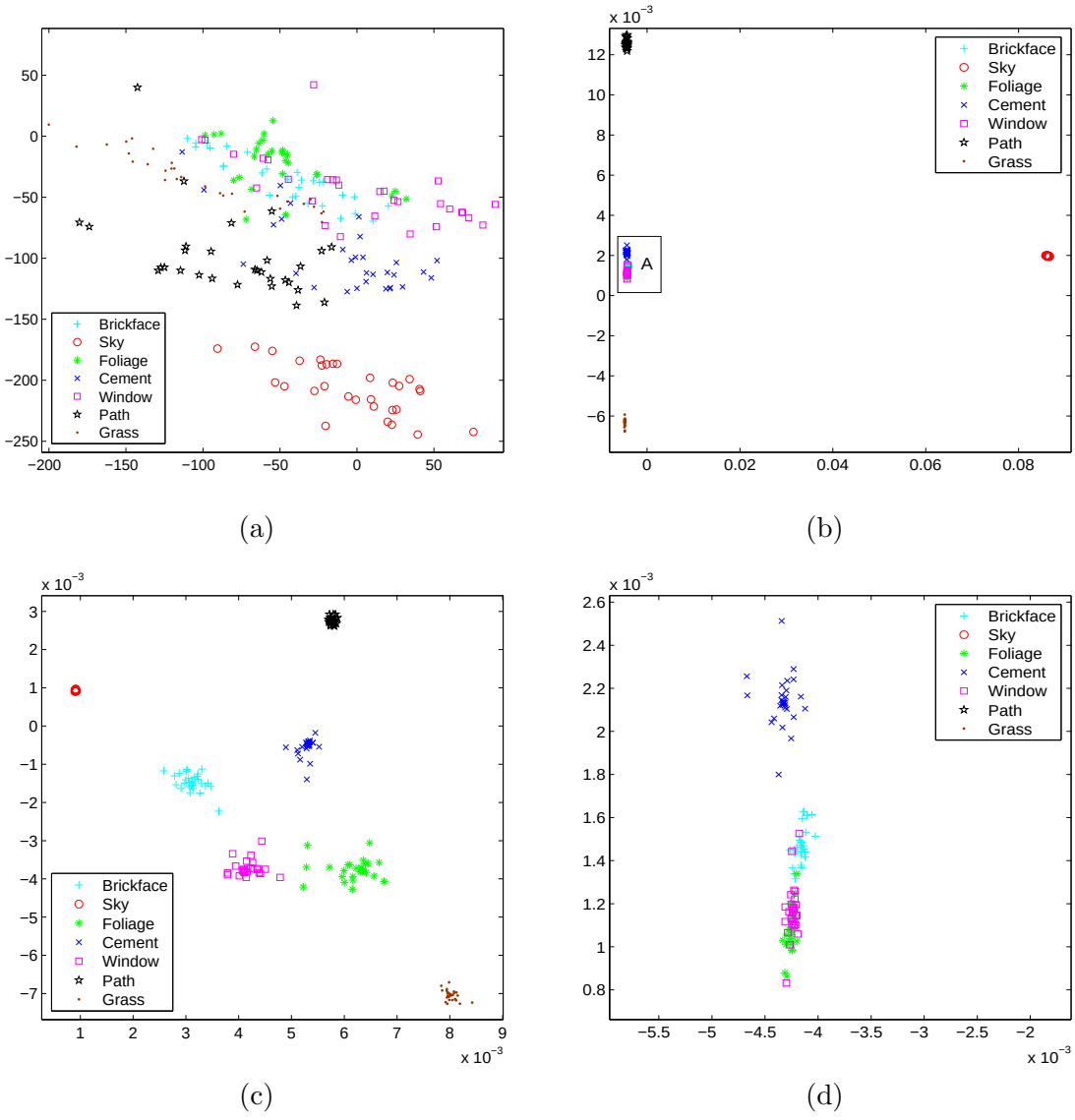

Fig. A.2. Distributions of 210 instances from the image segmentation database plotted based on the two most discriminant features obtained by three dimensionality reduction methods: (a) FLDA; (b) KDA; (c) KFDA; (d) magnification of region A in (b).

Fig. A.3. Thirty sample images of one subject from the YaleB face database used in the experiments.

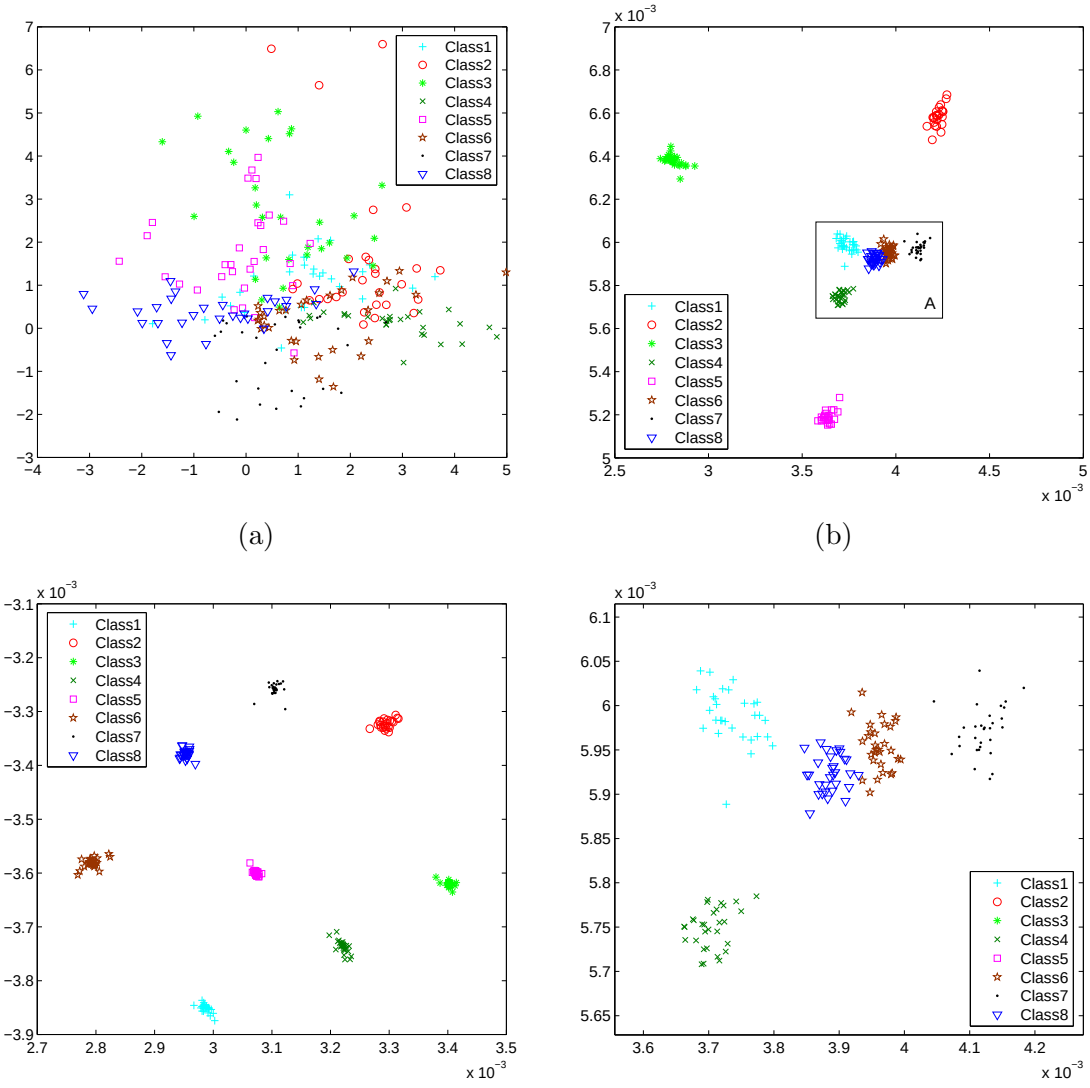

Fig. A.4. Distributions of 240 images from the YaleB face database plotted based on the two most discriminant features obtained by three dimensionality reduction methods: (a) DFLDA; (b) KDA; (c) KFDA; (d) magnification of region A in (b).

Fig. A.5. Some sample images from the FERET face database used in the experiments.

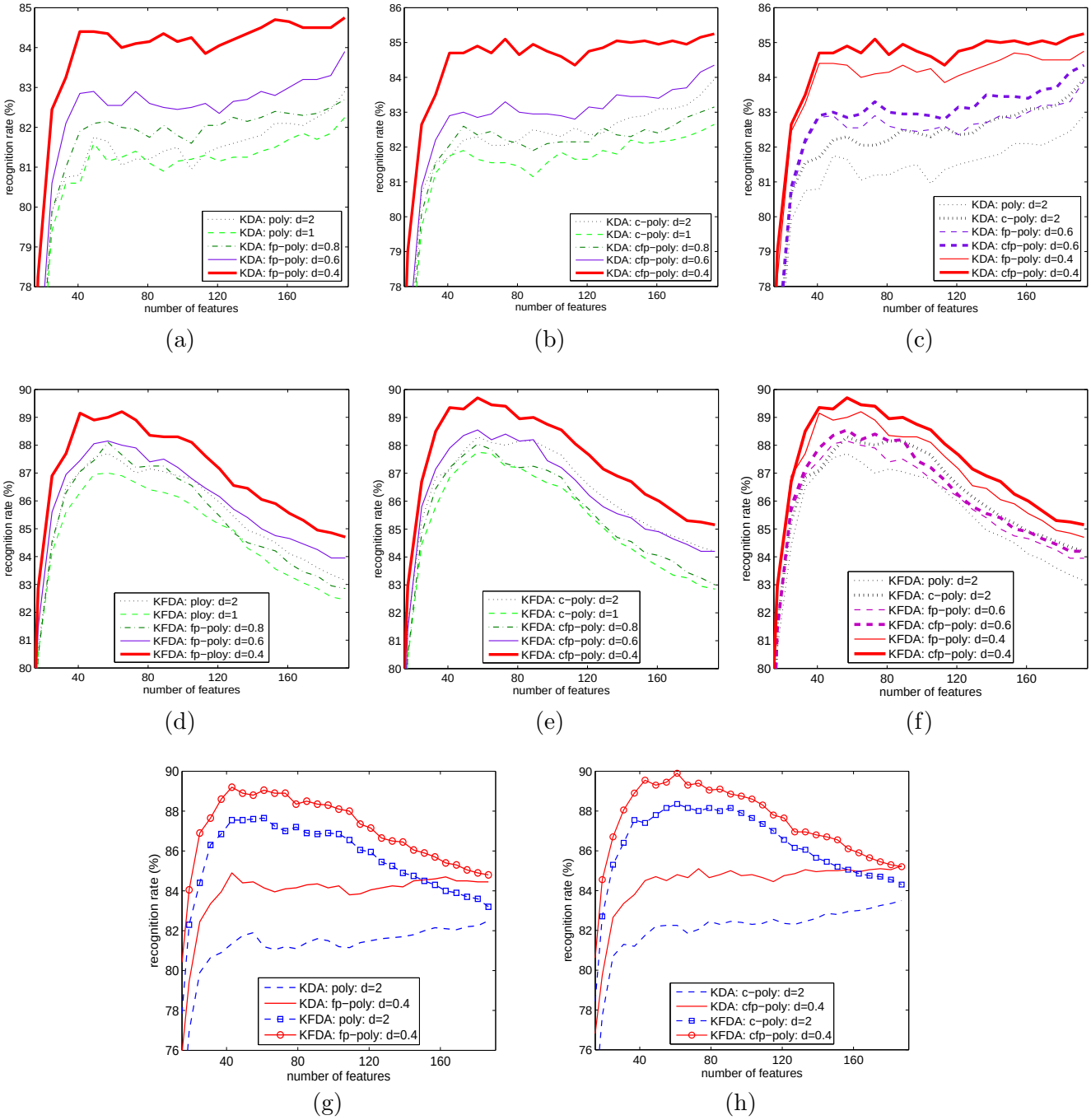

Fig. A.6. Comparison of KDA and KFDA on different non-stationary kernels. (a) KDA on polynomial kernel (poly) and fractional-power polynomial kernel (fp-poly); (b) KDA on cosine polynomial kernel (c-poly) and cosine fractional-power polynomial kernel (cfp-poly); (c) KDA on non-cosine kernels (poly and fp-poly) and cosine kernels (c-poly and cfp-poly); (d) KFDA on polynomial kernel (poly) and fractional-power polynomial kernel (fp-poly); (e) KFDA on cosine polynomial kernel (c-poly) and cosine fractional-power polynomial kernel (cfp-poly); (f) KFDA on non-cosine kernels (poly and fp-poly) and cosine kernels (c-poly and cfp-poly); (g) KDA vs. KFDA on polynomial kernel (poly) and fractional-power polynomial kernel (fp-poly); (h) KDA vs. KFDA on cosine polynomial kernel (c-poly) and cosine fractional-power polynomial kernel (cfp-poly).

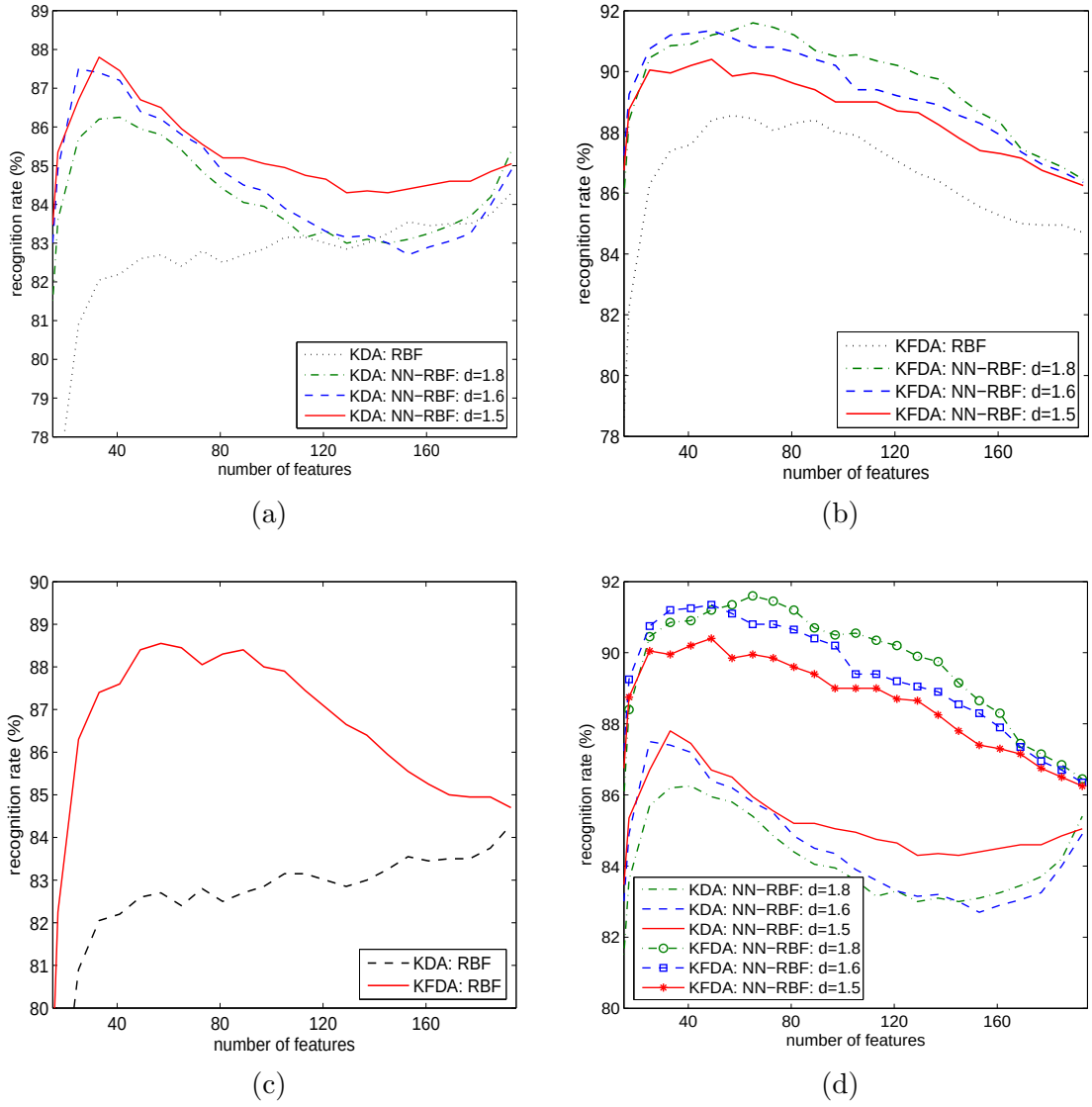

Fig. A.7. Comparison of KDA and KFDA on different isotropic kernels. (a) KDA on Gaussian RBF kernel (RBF) and non-normal Gaussian RBF kernel (NN-RBF); (b) KFDA on Gaussian RBF kernel (RBF) and non-normal Gaussian RBF kernel (NN-RBF); (c) KDA vs. KFDA on Gaussian RBF kernel (RBF); (d) KDA vs. KFDA on non-normal Gaussian RBF kernel (NN-RBF).

Table A.1
Best recognition rates (%) and corresponding numbers of features (·) for KPCA, KDA and KFDA on different kernels.

Algorithm	KPCA	KDA	KFDA		
Weighting function $w(d_{ij})$			d_{ij}^{-4}	d_{ij}^{-8}	d_{ij}^{-12}
poly $d = 2$	69.40(292)	83.30(198)	84.55(55)	86.70(59)	87.90(50)
poly $d = 1$	69.70(267)	82.45(199)	84.25(50)	86.45(56)	87.35(59)
c-poly $d = 2$	69.60(241)	84.20(199)	85.20(86)	87.40(75)	88.50(58)
c-poly $d = 1$	69.60(196)	82.90(197)	84.90(55)	86.85(53)	87.95(64)
fp-poly $d = 0.8$	69.70(269)	82.95(199)	84.85(59)	87.20(62)	88.15(53)
fp-poly $d = 0.6$	69.65(265)	84.05(199)	85.30(57)	87.45(63)	88.25(55)
fp-poly $d = 0.4$	69.65(267)	84.20(199)	86.65(81)	88.20(59)	89.70(63)
cfp-poly $d = 0.8$	69.65(208)	83.40(198)	85.30(55)	87.25(62)	88.25(56)
cfp-poly $d = 0.6$	69.70(292)	84.55(199)	86.00(51)	87.75(73)	88.60(51)
cfp-poly $d = 0.4$	69.70(292)	85.40(198)	86.95(67)	88.75(59)	90.00(62)
RBF $d = 2$	69.60(290)	84.80(199)	85.55(90)	87.85(55)	88.60(58)
NN-RBF $d = 1.8$	69.00(271)	86.65(38)	88.85(32)	90.65(61)	91.65(46)
NN-RBF $d = 1.6$	68.05(296)	87.80(27)	89.20(36)	90.70(31)	91.50(46)
NN-RBF $d = 1.5$	67.90(297)	87.90(37)	89.05(32)	90.05(38)	90.50(44)

Table A.2
Average error percentages (%) for KPCA and KDA when compared with KFDA on different kernels.

Algorithm/algorithm	KFDA/KPCA			KFDA/KDA		
Weighting function $w(d_{ij})$	d_{ij}^{-4}	d_{ij}^{-8}	d_{ij}^{-12}	d_{ij}^{-4}	d_{ij}^{-8}	d_{ij}^{-12}
poly $d = 2$	52.36	48.32	46.37	88.74	81.79	78.37
poly $d = 1$	54.50	50.58	48.98	90.08	83.56	80.63
c-poly $d = 2$	50.39	46.28	44.42	88.90	81.51	78.09
c-poly $d = 1$	53.51	49.43	47.68	90.56	83.53	80.44
fp-poly $d = 0.8$	52.82	48.93	47.39	90.47	83.62	80.84
fp-poly $d = 0.6$	50.72	46.86	45.45	90.30	83.25	80.62
fp-poly $d = 0.4$	46.66	43.68	42.29	90.93	84.93	81.99
cfp-poly $d = 0.8$	50.58	48.49	46.85	87.76	83.95	80.95
cfp-poly $d = 0.6$	49.46	45.94	44.57	90.41	83.82	81.16
cfp-poly $d = 0.4$	45.29	42.26	40.87	91.07	84.76	81.75
RBF $d = 2$	48.53	44.49	42.77	88.59	81.10	77.79
NN-RBF $d = 1.8$	39.64	34.45	32.25	82.66	71.78	67.06
NN-RBF $d = 1.6$	39.79	34.20	32.48	84.41	73.90	70.00
NN-RBF $d = 1.5$	39.31	35.61	34.06	90.23	81.71	78.10

About the Author – GUANG DAI received his B.Eng. degree in Mechanical Engineering from Dalian University of Technology and M.Phil. degree in Computer Science from Zhejiang University. He is currently a PhD student in the Department of Computer Science and Engineering at the Hong Kong University of Science and Technology. His main research interests include semi-supervised learning, nonlinear dimensionality reduction, and related applications.

About the Author – DIT-YAN YEUNG received his B.Eng. degree in Electrical Engineering and M.Phil. degree in Computer Science from the University of Hong Kong, and his Ph.D. degree in Computer Science from the University of Southern California in Los Angeles. He was an Assistant Professor at the Illinois Institute of Technology in Chicago before he joined the Department of Computer Science and Engineering at the Hong Kong University of Science and Technology, where he is currently an Associate Professor. His current research interests are in machine learning and pattern recognition.

About the Author – YUN-TAO QIAN received his B.E. and M.E. degrees in automatic control from Xi'an Jiaotong University in 1989 and 1992, respectively, and his Ph.D. degree in signal processing from Xidian University in 1996. From 1996 to 1998, he was a postdoctoral fellow at the Northwestern Polytechnical University. Since 1998, he has been with Zhejiang University where he is currently a Professor in Computer Science. His research interests include machine learning, signal and image processing, and pattern recognition.