KERNEL NEIGHBORHOOD PRESERVING EMBEDDING FOR CLASSIFICATION[1]

Tao Xiaoyan[*,**] Ji Hongbing[*] Men Jian[**]

[*](*School of Electronic Engineering, Xidian University, Xi'an 710071, China*)

[**](*The Telecommunication Engineering Institute, Air Force Engineering University, Xi'an 710077, China*)

Abstract The Neighborhood Preserving Embedding (NPE) algorithm is recently proposed as a new dimensionality reduction method. However, it is confined to linear transforms in the data space. For this, based on the NPE algorithm, a new nonlinear dimensionality reduction method is proposed, which can preserve the local structures of the data in the feature space. First, combined with the Mercer kernel, the solution to the weight matrix in the feature space is gotten and then the corresponding eigenvalue problem of the Kernel NPE (KNPE) method is deduced. Finally, the KNPE algorithm is resolved through a transformed optimization problem and QR decomposition. The experimental results on three real-world data sets show that the new method is better than NPE, Kernel PCA (KPCA) and Kernel LDA (KLDA) in performance.

Key words Kernel Neighborhood Preserving Embedding (KNPE); Neighborhood structure; Feature extraction; QR decomposition

CLC index TP391.4

DOI 10.1007/s11767-007-0215-9

I. Introduction

The problem called disaster of dimensionality appears in many fields including data mining, machine learning, computer vision and so on. For the high-dimensionality data, a commonly-used preprocessing step is to employ the dimensionality reduction method in order to find the lower "intrinsic dimensionality" hidden in the data. The classical method, such as Principle Component Analysis (PCA)[1] is designed to project the data points along the directions of maximal variances. When the class information is available, Linear Discriminant Analysis (LDA)[2] can be used to find an optimal projective space for discrimination. Being linear, both of them have the virtue of simple computations, while they donot apply to the nonlinear data. Meanwhile, several nonlinear methods based on manifold learning are proposed including Locally Linear Embedding (LLE)[3] and Laplacian eigenmap[4]. These nonlinear methods aim at preserving the intrinsic geometry of the local neighborhoods and do yield impressive results on some data set. However, their nonlinear property makes them computationally expensive. Moreover, since the exact nonlinear transformation model is unclear, the map for new test data canot be evaluated, thus limiting the application of these methods on the pattern recognition. To solve this problem, Xiaofei He, *et al.* propose some linear algorithms based on LLE, such as Locality Preserving Projections (LPP)[5] and Neighborhood Preserving Embedding (NPE)[6]. Experimental results on the face databases have shown that the features extracted by these algorithms have more discriminant power than those by PCA and LDA.

As nonlinear maps, Kernel based methods have also been considered, such as Kernel PCA (KPCA) and Kernel LDA (KLDA)[7]. Although these methods can capture the higher order relationships among data points, the structure of the manifold that data points may reside on is not considered. For this, the Kernel NPE (KNPE) method is presented in this paper, which aims at preserving the local manifold structure in the high-dimensional feature space. Given a set of data points in the data space, we first compute a weight matrix in the feature space. Then the eigenvalue problem of the

[1] Manuscript received date: December 6, 2007; revised date: June 20, 2008.

Communication author: Tao Xiaoyan, born in 1971, female, Ph.D. School of Electronic Engineering, Xidian University, Xi'an 710071, China.
Email: tao_xy@163.com.

standard NPE is kernelized. Finally an alternative formulation of KNPE is achieved based on a transformative optimization problem and QR decomposition, which can work regardless of the singular kernel matrix problem.

II. A Brief Review of Neighborhood Preserving Embedding

NPE is a linear approximation of the nonlinear LLE, aiming at preserving the local manifold structure. Given a set of points $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m\} \in \boldsymbol{R}^n$ and let $\boldsymbol{X} = [\boldsymbol{x}_1\ \boldsymbol{x}_2 \cdots \boldsymbol{x}_m]$. The algorithmic procedure is formally stated below:

Step 1 Constructing an adjacency graph

Let G denote a graph with m nodes and the i-th node corresponds to the data point $\boldsymbol{x}_i$. There are two ways to construct the graph:

(1) Unsupervised model Nodes are connected by a directed edge if $\boldsymbol{x}_j$ is among k nearest neighbors of $\boldsymbol{x}_i$.

(2) Supervised model An edge exists if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ belong to the same class. Notice that in this case one doesnot need to set the parameter k, the number of nearest neighbors, thus the method becomes fully automatic.

Step 2 Computing the weights

Let $\boldsymbol{W}$ be a weight matrix with W_{ij} having the weight of the edge from Node i to Node j, and 0 if there is no such edge. W_{ij} can be computed by minimizing the following object function:

$$\varepsilon(\boldsymbol{W}) = \sum_i \left| \boldsymbol{x}_i - \sum_j W_{ij} \boldsymbol{x}_j \right|^2 \tag{1}$$

The constraint is $\sum_j W_{ij} = 1$. The detailed process for solving this optimization problem can be referred to Ref.[3].

Step 3 Computing the projections

Consider the problem of mapping the data space to a line with $\boldsymbol{y} = [y_1\ y_2 \cdots y_m]^{\mathrm{T}}$ being such a map. A reasonable criterion for choosing a "good" map is to minimize the following cost function

$$\Phi(\boldsymbol{y}) = \sum_i \left(y_i - \sum_j W_{ij} y_j \right)^2 \tag{2}$$

under appropriate constraints. Suppose $\boldsymbol{y}^{\mathrm{T}} = \boldsymbol{a}^{\mathrm{T}} \boldsymbol{X}$, where $\boldsymbol{a}$ is the transformation vector. Then substitue this formular into Eq.(2), the new cost function is converted into

$$\boldsymbol{a}_{\mathrm{opt}} = \arg\min \boldsymbol{a}^{\mathrm{T}} \boldsymbol{X} \boldsymbol{M} \boldsymbol{X}^{\mathrm{T}} \boldsymbol{a}$$
$$\text{s.t.} \quad \boldsymbol{a}^{\mathrm{T}} \boldsymbol{X} \boldsymbol{X}^{\mathrm{T}} \boldsymbol{a} = 1$$

where $\boldsymbol{M} = (\boldsymbol{I} - \boldsymbol{W})^{\mathrm{T}} (\boldsymbol{I} - \boldsymbol{W})$. Finally, the basis vector of NPE is the eigenvector associated with the smallest eigenvalues of following generalized eigenvalue problem

$$\boldsymbol{X} \boldsymbol{M} \boldsymbol{X}^{\mathrm{T}} \boldsymbol{a} = \lambda \boldsymbol{X} \boldsymbol{X}^{\mathrm{T}} \boldsymbol{a} \tag{3}$$

Extending to the d-dimensionality space, the map of $\boldsymbol{X}$ is $\boldsymbol{Y} = [\boldsymbol{y}_1\ \boldsymbol{y}_2 \cdots \boldsymbol{y}_m]$, where $\boldsymbol{y}_i \in \boldsymbol{R}^d$. The solutions of Eq.(3) corresponding to the smallest d eigenvalues form the column vectors of the transformation matrix, namely $\boldsymbol{A} = [\boldsymbol{a}_1 \cdots \boldsymbol{a}_d]$.

From the analysis above, we can see that the standard NPE is a linear transformation and can apply to classification easily. However, NPE only uses the low dimension information in the data space, and when the higher order information can better represent the local structure, its performance will be affected. So, the kernel NPE is proposed.

III. Analysis on Kernel Neighborhood Preserving Embedding

1. Kernel neighborhood preserving embedding

For a given nonlinear mapping φ, the input data space can be mapped into the feature space $\boldsymbol{F}: \varphi: \boldsymbol{R}^n \to \boldsymbol{F}$. As a result, a sample in the original input space is mapped into a potentially much higher dimensional feature vector: $\boldsymbol{x} \to \varphi(\boldsymbol{x})$ in the feature space. Let $\boldsymbol{\varphi}(\boldsymbol{X})$ denote the data matrix in the feature space, $\boldsymbol{\varphi}(\boldsymbol{X}) = \left[\varphi(\boldsymbol{x}_1)\ \ \varphi(\boldsymbol{x}_2) \cdots \varphi(\boldsymbol{x}_m)\right]$.

As in NPE, consider first the construction of the adjacent graph. It is worth to note that the Euclidean distance between two data points in the feature space can be computed according to Ref.[8]

$$\left\| \boldsymbol{\varphi}(\boldsymbol{x}_i) - \boldsymbol{\varphi}(\boldsymbol{x}_j) \right\| = \sqrt{k(\boldsymbol{x}_i, \boldsymbol{x}_i) + k(\boldsymbol{x}_j, \boldsymbol{x}_j) - 2k(\boldsymbol{x}_i, \boldsymbol{x}_j)}$$

where $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\varphi}^{\mathrm{T}}(\boldsymbol{x}_i) \boldsymbol{\varphi}(\boldsymbol{x}_j)$.

Then the weight matrix can be achieved by minimizing

$$\varepsilon(\boldsymbol{W}) = \sum_i \left| \boldsymbol{\varphi}(\boldsymbol{x}_i) - \sum_j W_{ij} \boldsymbol{\varphi}(\boldsymbol{x}_j) \right| \tag{4}$$

An alternative expression for Eq.(4) is

$$\begin{aligned}\varepsilon(\boldsymbol{W}) &= \left\|\boldsymbol{\varphi}(\boldsymbol{X})\left(\boldsymbol{I}-\boldsymbol{W}^{\mathrm{T}}\right)\right\|_F^2 \\ &= \mathrm{tr}\left[(\boldsymbol{I}-\boldsymbol{W})\boldsymbol{\varphi}^{\mathrm{T}}(\boldsymbol{X})\boldsymbol{\varphi}(\boldsymbol{X})\left(\boldsymbol{I}-\boldsymbol{W}^{\mathrm{T}}\right)\right] \\ &= \mathrm{tr}\left[(\boldsymbol{I}-\boldsymbol{W})\boldsymbol{K}\left(\boldsymbol{I}-\boldsymbol{W}^{\mathrm{T}}\right)\right] \end{aligned} \tag{5}$$

where $\boldsymbol{K}=[k(\boldsymbol{x}_i,\boldsymbol{x}_j)]$ is the kernel matrix and $\boldsymbol{K}=\boldsymbol{\varphi}^{\mathrm{T}}(\boldsymbol{X})\boldsymbol{\varphi}(\boldsymbol{X})$. Since $\boldsymbol{K}$ is positive semi-definite, we have $\boldsymbol{K}=\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\mathrm{T}}=\left(\boldsymbol{U}\boldsymbol{\Lambda}^{1/2}\right)\left(\boldsymbol{U}\boldsymbol{\Lambda}^{1/2}\right)^{\mathrm{T}}$ with $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U}=\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix. Then Eq.(5) becomes

$$\begin{aligned}\varepsilon(\boldsymbol{W}) &= \mathrm{tr}\left[(\boldsymbol{I}-\boldsymbol{W})\left(\boldsymbol{U}\boldsymbol{\Lambda}^{1/2}\right)\left(\boldsymbol{U}\boldsymbol{\Lambda}^{1/2}\right)^{\mathrm{T}}\left(\boldsymbol{I}-\boldsymbol{W}^{\mathrm{T}}\right)\right] \\ &= \left\|(\boldsymbol{I}-\boldsymbol{W})\boldsymbol{U}\boldsymbol{\Lambda}^{1/2}\right\|_F^2 \\ &= \left\|\boldsymbol{\Lambda}^{1/2}\boldsymbol{U}^{\mathrm{T}}(\boldsymbol{I}-\boldsymbol{W})\right\|_F^2 \end{aligned} \tag{6}$$

So, $\boldsymbol{W}$ is constructed similarly as Eq.(1), but now $\boldsymbol{\Lambda}^{1/2}\boldsymbol{U}^{\mathrm{T}}$ replaces $\boldsymbol{X}$.

Finally the eigenvalue problem in the feature space can be written as follows

$$\boldsymbol{\varphi}(\boldsymbol{X})\boldsymbol{M}\boldsymbol{\varphi}^{\mathrm{T}}(\boldsymbol{X})\boldsymbol{v}=\lambda\boldsymbol{\varphi}(\boldsymbol{X})\boldsymbol{\varphi}^{\mathrm{T}}(\boldsymbol{X})\boldsymbol{v} \tag{7}$$

Since the eigenvectors of Eq.(4) can be expressed by a linear combinations of $\boldsymbol{\varphi}(\boldsymbol{x}_1)$, $\boldsymbol{\varphi}(\boldsymbol{x}_2)$, $\cdots$, $\boldsymbol{\varphi}(\boldsymbol{x}_m)$, there exist coefficients α_i, $i=1,2,\cdots,m$, such that

$$\boldsymbol{v}=\sum_{i=1}^{m}\alpha_i\boldsymbol{\varphi}(\boldsymbol{x}_i)=\boldsymbol{\varphi}(\boldsymbol{X})\boldsymbol{\alpha} \tag{8}$$

where $\boldsymbol{\alpha}=[\alpha_1\ \alpha_2\ \cdots\ \alpha_m]^{\mathrm{T}}\in\boldsymbol{R}^m$.

Substituting Eq.(8) into Eq.(7), we can finally obtain the following eigenvector problem

$$\boldsymbol{KMK\alpha}=\lambda\boldsymbol{KK\alpha} \tag{9}$$

2. Computations of the projection

It easily can be seen that Eq.(9) canot be resolved directly and some transformation is needed. From Ref.[9], Eq.(9) can be converted into the following optimal problem

$$\boldsymbol{\alpha}=\mathrm{argmin}\frac{\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{KMK\alpha}}{\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{KK\alpha}} \tag{10}$$

For convenience, Set $J(\boldsymbol{\alpha})=\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{KMK\alpha}/(\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{KK}\cdot\boldsymbol{\alpha})$, which evaluates the neighborhood preserving power of the projective map $\boldsymbol{\alpha}$.

First, let us consider the QR decomposition of the kernel matrix again. Suppose $t=\mathrm{rank}(\boldsymbol{K})$ and $\boldsymbol{\gamma}_1,\boldsymbol{\gamma}_2,\cdots,\boldsymbol{\gamma}_t$ are $\boldsymbol{K}$'s orthogonal eigenvectors corresponding to t nonzero eigenvalues. Then, $\boldsymbol{K}$ can be expressed by $\boldsymbol{K}=\boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}^{\mathrm{T}}$, where $\boldsymbol{P}=[\boldsymbol{\gamma}_1\ \boldsymbol{\gamma}_2\ \cdots\ \boldsymbol{\gamma}_t]$ and $\boldsymbol{\Lambda}$ is the corresponding eigenvalue matrix.

Substituting $\boldsymbol{K}$ into Eq.(10), we have

$$J(\boldsymbol{\alpha})=\frac{\left(\boldsymbol{\Lambda}^{1/2}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{\alpha}\right)^{\mathrm{T}}\left(\boldsymbol{\Lambda}^{1/2}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{MP}\boldsymbol{\Lambda}^{1/2}\right)\left(\boldsymbol{\Lambda}^{1/2}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{\alpha}\right)}{\left(\boldsymbol{\Lambda}^{1/2}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{\alpha}\right)^{\mathrm{T}}\boldsymbol{\Lambda}\left(\boldsymbol{\Lambda}^{1/2}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{\alpha}\right)} \tag{11}$$

Let $\boldsymbol{\beta}=\boldsymbol{\Lambda}^{1/2}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{\alpha}$. Then Eq.(11) can be reduced to

$$J(\boldsymbol{\beta})=\frac{\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{\beta}}{\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{T}\boldsymbol{\beta}} \tag{12}$$

where $\boldsymbol{B}=\boldsymbol{\Lambda}^{1/2}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{MP}\boldsymbol{\Lambda}^{1/2}$ and $\boldsymbol{T}=\mathrm{diag}\{\lambda_1,\lambda_2,\cdots,\lambda_t\}$.

And now $\boldsymbol{T}$ is positive definite and $\boldsymbol{B}$ is positive semi-definite. So Eq.(9) is now a standard generalized eigenvalue problem and the solutions $\boldsymbol{\beta}_j$ are the eigenvectors corresponding to d smallest eigenvalues. Then we can obtain a set of optimal solutions $\alpha_j=\boldsymbol{P}\boldsymbol{\Lambda}^{1/2}\boldsymbol{\beta}_j$ $(j=1,2,\cdots,d)$ according to $\boldsymbol{\beta}=\boldsymbol{\Lambda}^{1/2}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{\alpha}$. Thereby, the optimal projective vectors in the feature space are

$$\boldsymbol{v}_j=\boldsymbol{\varphi}(\boldsymbol{X})\alpha_j=\boldsymbol{\varphi}(\boldsymbol{X})\boldsymbol{P}\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\beta}_j,\ j=1,2,\cdots,d \tag{13}$$

Then the projections $\boldsymbol{z}$ of arbitrary $\boldsymbol{\varphi}(\boldsymbol{x})$ in feature space can be obtained by the following transformation

$$\boldsymbol{z}=\boldsymbol{V}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x}) \tag{14}$$

where

$$\begin{aligned}\boldsymbol{V} &= [\boldsymbol{v}_1\ \boldsymbol{v}_2\ \cdots\ \boldsymbol{v}_d] \\ &= [\boldsymbol{\varphi}(\boldsymbol{X})\boldsymbol{P}\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\beta}_1\ \boldsymbol{\varphi}(\boldsymbol{X})\boldsymbol{P}\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\beta}_2\ \cdots \\ &\quad\ \boldsymbol{\varphi}(\boldsymbol{X})\boldsymbol{P}\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\beta}_d] \\ &= (\boldsymbol{\varphi}(\boldsymbol{X})\boldsymbol{P}\boldsymbol{\Lambda}^{-1/2})[\boldsymbol{\beta}_1\ \boldsymbol{\beta}_2\ \cdots\ \boldsymbol{\beta}_d]\end{aligned}$$

Thus the final projection is

$$\begin{aligned}\boldsymbol{z} &= \left[\boldsymbol{\beta}_1\ \boldsymbol{\beta}_2\ \cdots\ \boldsymbol{\beta}_d\right]^{\mathrm{T}}\boldsymbol{\Lambda}^{-1/2}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{\varphi}^{\mathrm{T}}(\boldsymbol{X})\boldsymbol{\varphi}(\boldsymbol{x}) \\ &= \left[\boldsymbol{\beta}_1\ \boldsymbol{\beta}_2\ \cdots\ \boldsymbol{\beta}_d\right]^{\mathrm{T}}\boldsymbol{\Lambda}^{-1/2}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{k}\end{aligned}$$

where $\boldsymbol{k}$ is an m-vector of inner products of $\boldsymbol{x}$ with the data in feature space: $(\boldsymbol{k})_n=k(\boldsymbol{x}_n,\boldsymbol{x})$. Consequently, Kernel Neighborhood Preserving Embedding (KNPE) algorithm can be described as

the following.

Algorithm KNPE

Given a data matrix $\boldsymbol{X}=[\boldsymbol{x}_1\ \boldsymbol{x}_2\ \cdots\ \boldsymbol{x}_m]$ with a kernel function k

Step 1 Compute matrix $\boldsymbol{W}$ according to Eq.(6)

Step 2 Compute $\boldsymbol{K}=[k(\boldsymbol{x}_i,\boldsymbol{x}_j)]$

Step 3 Compute the QR decomposition of $\boldsymbol{K}$ as $\boldsymbol{K}=\boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}^{\mathrm{T}}$

Step 4 Suppose $\boldsymbol{B}=\boldsymbol{\Lambda}^{1/2}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{P}\boldsymbol{\Lambda}^{1/2}$ and $\boldsymbol{T}=$ $=\mathrm{diag}\{\lambda_1,\lambda_2,\cdots,\lambda_t\}$, then Compute the eigenvectors of $\boldsymbol{T}^{-1}\boldsymbol{B}$ corresponding to d smallest eigenvalues

Step 5 For any input data $\boldsymbol{x}$, compute its projection $\boldsymbol{z}=[\boldsymbol{\beta}_1\ \boldsymbol{\beta}_2\ \cdots\ \boldsymbol{\beta}_d]^{\mathrm{T}}\boldsymbol{\Lambda}^{-1/2}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{k}$.

IV. Experimental Results and Analysis

In this section, we conduct experiments to demonstrate the effectiveness of the proposed KNPE algorithm in comparison with other kernel-based nonlinear feature extraction algorithms as well as NPE. In all experiments, for simplicity, we apply the nearest-neighbor classifier in the reduced dimensional space.

1. Sonar data

First, Sonar data[10] is selected to test the performance of the proposed algorithm. This data set has 208 data, which is made up of 2 classes with 111 data and 97 data respectively, and each datum consists of 60 attributes. We randomly split the data into equally sized training and test sets with the Gaussian RBF kernel: $\exp(-\|\boldsymbol{x}-\boldsymbol{y}\|^2/2\sigma^2)$ used. We compare KNPE with NPE and KPCA. The random split is repeated 10 times and the average prediction error rates are gotten. For fair comparison, parameter σ is chosen from 2^{-2} to 2^{-20} for each kernel method and the best results are reported. Fig.1 plots the error rates versus dimensionality reduction. We can see that KNPE performs the best, while NPE performs poorly. And the performance appeared ditherring when the dimensionality changed, which is the result of that the projective vectors gotten by all of these methods are not orthogonal.

2. Yale face database

The Yale face database[11] was constructed at the Yale Center for Computational Vision and Control. It contains 165 gray scale images of 15 individuals. The images demonstrate variations in lighting condition, facial expression (normal, happy, sad, sleepy, surprised, and wink). Each image is cropped into 32×32 pixels. A subset with l (=3, 4, 5) images per individual is randomly sampled to form the training set. Again the Gaussian kernel parameter is chosen from 2^{-20} to 2^{-30}. We average the results over 10 random splits. The performance of KNPE is compared with NPE, KPCA and KLDA (KPCA+LDA) proposed by Yang, *et al.* In KLDA, for simplicity, we set the reduced dimension to the number of the nonzero eigenvalues in the first phase (KPCA). And there are at most c –1 nonzero generalized eigenvalues for KLDA, where c is the number of individuals[7]. Fig.2 plots the results. As can be seen, our algorithm performs the best. Surprisingly, NPE performs comparatively to KNPE while KPCA and KLDA perform poorly, which shows that the local neighborhood information has more recognition power than the overall nonlinear information. It is also interesting to see that KLDA outperforms KPCA since the category information is considered in KLDA method.

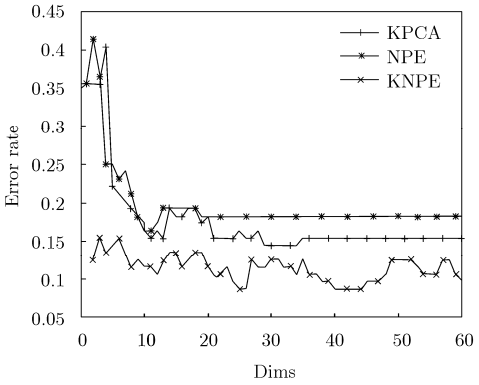
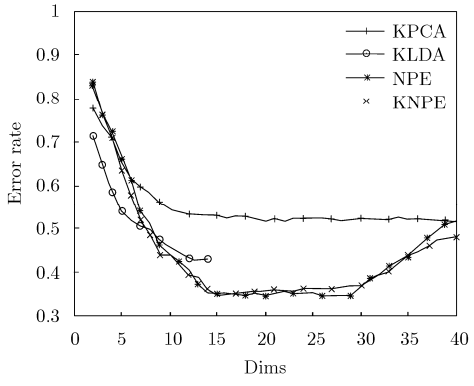
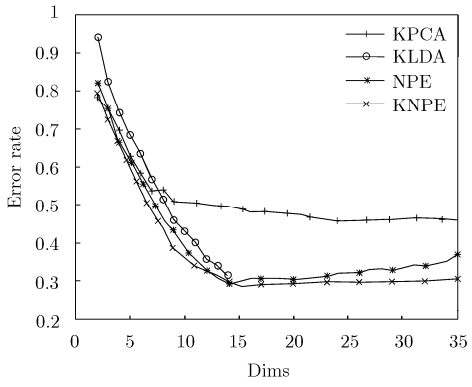


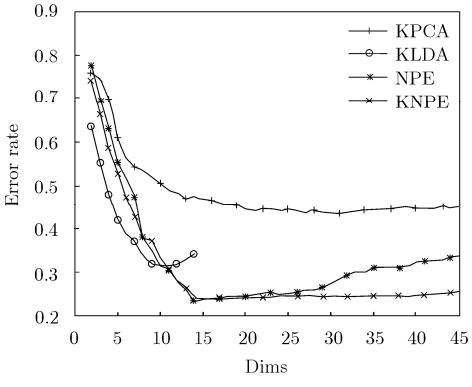
Fig.1 Error rate vs. dimensionality reduction on Sonar dataset



(a) l=3



(b) l=4



(c) l=5

Fig.2 Error rate vs. dimensionality reduction on Yale face database

3. Radar dataset

Radar target recognition refers to the detection and recognition of target signatures using high-resolution range profiles, in our case, in inverse synthetic aperture radar. Each datum in this data set represents a one-dimensional image of some plane, called a range profile. There are 3 different plane models, *i.e.* J-6, J-7 and F-1, acquired in a microwave anechoic chamber. The dimensionality of the range profiles is 64. The full data set has 770 data, and one third data is randomly selected as the training set. We use the same experimental procedure as in the previous section. The results in Fig.3 show that KNPE outperforms NPE and KPCA, while performed comparably to KLDA. Different from the results on Yale face database, NPE performs the worst, which shows that the nonlinear overall information play more role for this data set. For KLDA, because the class number is 3, the reduced dimensionality is reduced to 2. And the better result at Dims=2 is plotted in this figure.

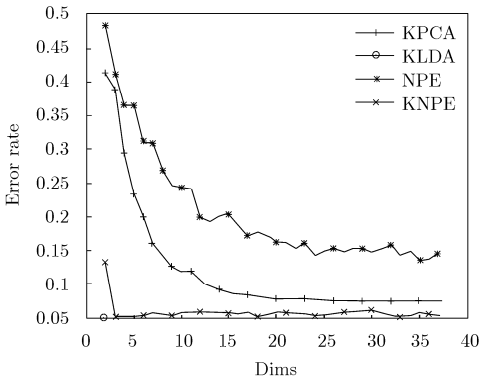


Fig.3 Error rate vs. dimensionality reduction on radar data set

V. Conclusion

In this paper, we show that NPE, a simple yet powerful feature extraction method, can be efficiently kernelized. This not only maintains the ability of NPE to preserve the local manifold structure, but also takes the higher order information into account. Besides, the proposed resolving procedure based on an alternative optimization formula and QR decomposition has settled the singular matrix problem. The experimental results indicate the effectiveness of KNPE.

References

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

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

[3] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, **290**(2000)5500, 2323–2326.

[4] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. Advances in Neural Information Processing Systems 14, Cambridge, MA, Dec. 3–8, 2001, 585–591.

[5] X. F. He, S. C. Yan, Y. X. Hu, *et al.* Face recognition using Laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* **27**(2005)3, 328–340.

[6] X. F. He, D. Cai, S. C. Yan, *et al.* Neighborhood preserving embedding. Proc. of the 10th IEEE In-

ternational Conference on Computer Vision, Beijing, Oct. 15–21, 2005, 1208–1213.

[7] M. H. Yang. Kernel eigenfaces vs. kernel fisherfaces: face recognition using kernel methods. Proc. of Fifth International Conference Automatic Face and Gesture Recognition, Washington DC, May 20–21, 2002, 215–220.

[8] L. Zhang, W. D. Zhou, and L. C. Jiao. Kernel clustering algorithm. *Chinese Journal of Computers*, **25**(2002)6, 587–590 (in Chinese).

张莉, 周伟达, 焦李成. 核聚类算法. 计算机学报, **25**(2002)6, 587–590.

[9] D. Cai, X. F. He, J. W. Han, *et al*. Orthogonal Laplacianfaces for face recognition. *IEEE Transactions on Image Processing*, **15**(2006)11, 3608–3614.

[10] M. Murphy. UCI-benchmark repository of artificial and real data sets. http://www.ics.uci.edu/~mlearn. July 2007.

[11] D. Cai. Face databases. http:// www.ews.uiuc.edu/ ~dengcai2. July 2007.