International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:4, 2007

An Iterative Algorithm for KLDA Classifier

D.N. Zheng, J.X. Wang, Y.N. Zhao, and Z.H. Yang

***Abstract*—**The Linear discriminant analysis (LDA) can be generalized into a nonlinear form — kernel LDA (KLDA) expediently by using the kernel functions. But KLDA is often referred to a general eigenvalue problem in singular case. To avoid this complication, this paper proposes an iterative algorithm for the two-class KLDA. The proposed KLDA is used as a nonlinear discriminant classifier, and the experiments show that it has a comparable performance with SVM.

***Keywords*—**Linear discriminant analysis (LDA), kernel LDA (KLDA), conjugate gradient algorithm, nonlinear discriminant classifier.

I. Introduction

SINCE the support vector machine (SVM) introduced a general kernel method, which can transform the input space to a higher dimensional feature space via an implicit nonlinear mapping function, many linear methods can be generalized into their nonlinear forms by the kernel trick, such as kernel principle component analysis (KPCA) [1], kernel linear discriminant analysis (KLDA) [2], [3], kernel fisher discriminant (KFD) [4], etc.

The traditional LDA can find the optimal projection to preserve the cluster structure in linearly separable data, while KLDA can overcome the limitation due to non-linearly separable data. The optimal solution for KLDA is obtained by solving a general eigenvalue problem, but the within-class scatter matrix is often singular. The authors in [3] recommended solving this difficulty by generalized singular value decomposition. The authors in [4] added a multiple of the identity matrix to the within-class scatter matrix, and made it become positive definite.

In this paper, a fast and stable iterative algorithm for KLDA in the two-class case is proposed to avoid the eigenvalue decomposition in singular case. The iteration procedure based on the conjugate gradient algorithm converges very fast and stably. The proposed KLDA is used as a two-class classifier like SVM. And it is compared with SVM classifier in the experiments.

Manuscript received December 24, 2004.

D.N. Zheng is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (phone: 86-10-62775613; e-mail: zdn02@mails.tsinghua.edu.cn).

J.X. Wang, Y.N. Zhao, and Z.H. Yang, are with the professors of the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China. (e-mail: wjx@mail.tsinghua.edu.cn, zhaoyn@mail.tsinghua.edu.cn, yzh@mail.tsinghua.edu.cn).

II. KLDA in Singular Case

Using kernel functions, the linear discriminant analysis (LDA) can be generalized to nonlinear discriminant analysis. The nonlinear decision function in the input space is equivalent to a linear decision function in the transformed space implied by the kernel functions [2].

Let $X = X_1 \bigcup X_2 = \{x_1, \cdots, x_{n_1}, x_{n_1+1}, \cdots, x_{n_1+n_2}\}$ be a set of training vectors of two classes, where n_1 and n_2 denote the number of samples in the two classes X_1 and X_2. Suppose that the input space $\mathcal{X}$ is transformed into a Hilbert space $\mathcal{F}$ by a nonlinear mapping function $\phi : x \to \phi(x)$. And the between-class scatter matrix S_b and the within-class scatter matrix S_w in the space $\mathcal{F}$ can be defined by [3]

$$\begin{aligned} S_b &= H_b H_b^T, \\ H_b &= [\sqrt{n_1}(\bar{\phi}_1 - \bar{\phi}), \sqrt{n_2}(\bar{\phi}_2 - \bar{\phi})] \end{aligned} \tag{1}$$

and

$$\begin{aligned} S_w &= H_w H_w^T, \\ H_w &= [\phi(x_1) - \bar{\phi}_1, \cdots, \phi(x_{n_1}) - \bar{\phi}_1, \phi(x_{n_1+1}) - \bar{\phi}_2, \cdots, \\ &\quad \phi(x_{n_1+n_2}) - \bar{\phi}_2] \end{aligned} \tag{2}$$

where $\bar{\phi}_1 = \frac{1}{n_1}\sum_{i=1}^{n_1}\phi(x_i), \bar{\phi}_2 = \frac{1}{n_2}\sum_{i=n_1+1}^{n_1+n_2}\phi(x_i), \bar{\phi} = \frac{1}{n_1+n_2}\sum_{i=1}^{n_1+n_2}\phi(x_i)$ denote the mean of class 1, the mean of class 2 and the mean of entire data, respectively.

The linear decision function for the two classes in $\mathcal{F}$ is $f(x) = w^T\phi(x) + b$, and w can be obtained by solving a general eigenvalue problem

$$S_b w = \lambda S_w w \tag{3}$$

The eigenvectors are linear combinations of $\mathcal{F}$ elements [1], and all solutions w with nonzero eigenvalues lie in the span of $\{\phi(x_1), \cdots, \phi(x_{n_1+n_2})\}$. Thus there exist coefficients α_i (i=1,$\cdots$, n_1+n_2) that satisfy,

$$w = \sum_{i=1}^{n_1+n_2}\alpha_i\phi(x_i) = Y\alpha \tag{4}$$

where $Y = [\phi(x_1), \cdots, \phi(x_{n_1+n_2})]$, $\alpha = [\alpha_1, \cdots, \alpha_{n_1+n_2}]^T$. Substitute w with (4) and left multiply it by Y^T, and the equation (3) can be rewritten as follows

$$\begin{aligned} Y^T S_b Y\alpha &= \lambda Y^T S_w Y\alpha \\ (Y^T H_b)(Y^T H_b)^T\alpha &= \lambda(Y^T H_w)(Y^T H_w)^T\alpha \end{aligned} \tag{4}$$

Now α is the eigenvector with the largest eigenvalue in (4). Using kernel operator $K_{ij} = \phi(x_i)^T\phi(x_j) = k(x_i, x_j)$, the

matrixes $Y^T H_b$ and $Y^T H_w$ can be computed by

$$(Y^T H_b)_{ij} = \phi(x_i)^T \sqrt{n_j}(\bar{\phi}_j - \bar{\phi}) = \sqrt{n_j}(\frac{1}{n_j}\sum_{x_k \in X_j} K_{ik} - \frac{1}{n_1+n_2}\sum_{x_k \in X} K_{ik}) \quad (5)$$

for i=1,⋯,n_1+n_2; j=1,2, and

$$(Y^T H_w)_{ij} = \phi(x_i)^T(\phi(x_j) - \bar{\phi}_r) = K_{ij} - \frac{1}{n_r}\sum_{x_k \in X_r} K_{ik} \quad (6)$$

for i=1,⋯,n_1+n_2; r=1, j=1,⋯,n_1; r=2, j=n_1+1,⋯,n_1+n_2. To simplify the denotation, we let $T_b = (Y^T H_b)(Y^T H_b)^T$, $T_b = (Y^T H_b)(Y^T H_b)^T$. Because $T_w \in R^{(n_1+n_2)\times(n_1+n_2)}$ is generally a singular matrix, α cant be computed by applying eigenvalue decomposition to $T_w^{-1} T_b$.

III. Proposed Iterative Algorithm

In order to overcome the complication of a singular T_w, we propose an iterative optimization algorithm to realize the nonlinear discriminant classifier.

The classical Fisher criterion function is to maximize the ratio of the between-class scatter of the projected samples to the within-class scatter of the projected samples [5], i.e,

$$J(w) = \max_w \frac{w^T S_b w}{w^T S_w w} \quad (7)$$

Substitute w with (4), and (7) becomes a function of α

$$J(\alpha) = \max_\alpha \frac{\alpha^T Y^T S_b Y \alpha}{\alpha^T Y^T S_w Y \alpha} = \max_\alpha \frac{\alpha^T T_b \alpha}{\alpha^T T_w \alpha} \quad (8)$$

It is a maximization problem. We can use an iterative algorithm − conjugate gradient algorithm to find the maximum extremum of this criterion function.

The gradient of $J(\alpha)$ at $\alpha(k)$ is

$$\nabla J(\alpha(k)) = \frac{2[(\alpha(k)^T T_w \alpha(k))T_b - (\alpha(k)^T T_b \alpha(k))T_w]\alpha(k)}{(\alpha(k)^T T_w \alpha(k))^2} \quad (9)$$

And the iterative algorithm is as follows:

1. Compute the kernel matrix K;
2. Compute the matrixes $Y^T H_b$, $Y^T H_w$, and the scatter matrixes T_b, T_w;
3. Initialize $\alpha(0)$ by a random vector, and normalize it;
$$\alpha(0) = \alpha(0)/\sqrt{\alpha(0)^T K \alpha(0)}$$
4. Compute the gradient $\nabla J(\alpha(0))$;
5. Compute the initial search direction;
$$s(0) = \nabla J(\alpha(0))/\left\|\nabla J(\alpha(0))\right\|$$
6. Set the initial step length $\rho(0)=[c,\cdots,c]^T$, where c>0, and set the iteration number k=0;
7. While k<N do
8. Update $\alpha_i(k+1) = \alpha_i(k) + \rho_i(k)s_i(k)$, i=1,⋯,n_1+n_2, and normalize $\alpha(k+1) = \alpha(k+1)/\sqrt{\alpha(k+1)^T K \alpha(k+1)}$;
9. Compute the gradient $\nabla J(\alpha(k+1))$;
10. Update
$s(k+1) = \nabla J(\alpha(k+1)) + s(k)\left\|\nabla J(\alpha(k+1))\right\|^2 / \left\|\nabla J(\alpha(k))\right\|^2$
and normalize $s(k+1) = s(k+1)/\left\|s(k+1)\right\|$;
11. Update
$\rho_i(k) = \rho_i(k)a^t$, $t = sign(\nabla J_i(\alpha(k))\nabla J_i(\alpha(k+1)))$,
where a>1, i=1,⋯,n_1+n_2;
12. If ||α(k+1)-α(k)||<ε$_1$ and |J(α(k+1))-J(α(k))|<ε$_2$, then stop the iteration;
13. $k = k$+1;
14. End

The w is a unit vector in $\mathcal{F}$, i.e. $w^T w = \alpha^T Y^T Y \alpha = \alpha^T K \alpha$ =1, so α is normalized by $\alpha = \alpha/\sqrt{\alpha^T K \alpha}$ in Step 3 and 8. The step length ρ(k) is adapted to each iteration by dynamic modification. Its initial value ρ(0) in Step 6 can be small to make a stable iteration. The update step a in Step 11 can be chosen from 1<a<2. The Step 12 checks whether the iteration can be terminated successfully.

After α obtained, the nonlinear decision function in the input space $\mathcal{X}$ can be defined by

$$f(x) = \sum_{i=1}^{n_1+n_2} \alpha_i k(x_i, x) + b \quad (10)$$

where b determines the offset. Use

$$m_1 = \frac{1}{n_1}\sum_{j=1}^{n_1}\sum_{i=1}^{n_1+n_2} \alpha_i K_{ij} \text{ and } m_2 = \frac{1}{n_2}\sum_{j=n_1+1}^{n_1+n_2}\sum_{i=1}^{n_1+n_2} \alpha_i K_{ij} \quad (11)$$

to denote the class means of the projected samples, and set b as $b_1 = -(m_1+m_2)/2$, $b_2 = -(n_1 m_1 + n_2 m_2)/(n_1+n_2)$, etc. If m_1>m_2 and min_1, max_2 denote the minimum projection of Class 1 and the maximum projection of Class 2, then the offset $b_3 = -(\min_1 + \max_2)/2$ will give a maximal margin between two separable classes.

For m classes classification, we can use m decision functions to separate them by one-against-all, or use m(m-1)/2 decision functions to separate them by one-against-one.

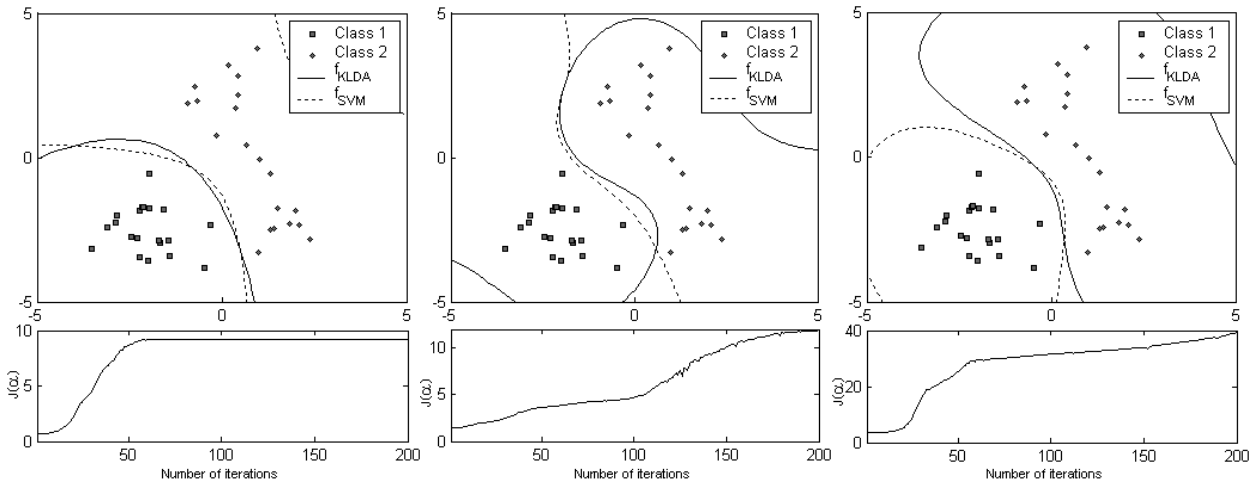
IV. Experimental Results

A. Synthetic Data

We first perform our algorithm on some synthetic data to illuminate its convergent speed and behavior according to the choice of the kernel function.

The Class 1 is a set of 20 points (x,y), which are generated by two independent variables such that $X \sim N(-2,1)$, $Y \sim N(-2,1)$. The Class 2 has 20 points too. Half of them are generated by $X \sim N(0,1)$ and $Y \sim N(2,1)$, and the other half are generated by $X \sim N(2,1)$ and $Y \sim N(-2,1)$. For a comparison purpose, the decision functions of KLDA (b=b_1) and SVM with the different kernel functions (polynomial kernel $k(x, x_i) = (x \cdot x_i + 1)^p$, RBF kernel $k(x, x_i) = \exp(-\left\|x - x_i\right\|^2)/(2\sigma^2))$) are all shown in Fig.1. The SVM classifiers make the maximal margins between the two classes, whereas the KLDA classifiers give small

scatters to the sample projections of each class and large scatters to those of different classes. And this is represented in the middle case (p=3) in Fig.1 evidently. The proposed algorithm converges very fast (c=10^{-4} in Step 6, a=1.2 in Step 11), and the results are not sensitive to the initial value α(0).



(a) polynomial kernel (p=2) (b) polynomial kernel (p=3) (c) RBF kernel (σ=2)

Fig.1 (Top) The decision functions of KLDAs and SVMs; (Bottom) The criterion function J(α) against the number of iterations.

B. AR Face Database

In the second experiment, we test the KLDA and SVM methods on the AR face database [5]. We select 10 different individuals randomly from this database. Each individual has 26 images. All the 260 images are full cropped into the same size 90×120. We first project the image data into an 80-dimensional PCA subspace, and then use the first 13 images of each person for training and the last 13 for testing.

The recognition rates obtained by KLDA and SVM using one-against-all are listed in Table I. This table shows that it's better for KLDA to choose b=b_2 for polynomial kernel and b=b_3 for RBF kernel. The training time and testing time of the two methods are near equal in this experiment.

TABLE I
RECOGNITION RATES OBTAINED BY KLDAS AND SVM ON THE AR FACE IMAGES

Method	Kernel function		
	Polynomial $p=2$	Polynomial $p=3$	RBF $\sigma=20$
KLDA ($b=b_1$)	83.1%	86.2%	90.0%
KLDA ($b=b_2$)	87.7%	90.8%	88.5%
KLDA ($b=b_3$)	83.8%	87.7%	90.8%
SVM	89.2%	91.5%	90.0%

C. Benchmark Repository

In the third experiment, the results are obtained on the Benchmark Repository used in [4] and [7]. The Benchmark Repository consist of 13 artificial and real world data sets: Banana, Breast Cancer, Diabetis, Flare-Solar, German, Heart, Image, Ringnorm, Splice, Thyroid, Titanic, Twonorm and Waveform, from the UCI, DELVE and STATLOG benchmark repositories. Each data set is partitioned as a binary classify-cation problem, and 100 partitions into test and training set were generated. We only select the first partitions of the 13 sets in this experiment.

We compared KLDA with SVM both using RBF kernel function, and the parameter σ is found by minimizing the error rates of classification. The parameter C in SVM is fixed at 10. The test error rates on the 13 data sets and the values of σ are tabulated in Table II. From this table, we can see that: the KLDA obtained by the iterative algorithm is competitive to SVM on almost all data sets (slightly better in 4 cases and slightly worse in 4 cases); the offset b for KLDA (see (10)) determines the performance of KLDA, and it should be well estimated. In this experiment, the b_1, b_2 and b_3 are still used. But the parameter b can be also optimized to minimize the test errors.

TABLE II
ERROR RATES OBTAINED BY KLDAS AND SVM USING RBF KERNEL ON THE BENCHMARK REPOSITORY.

	σ	KLDA ($b=b_1$)	KLDA ($b=b_2$)	KLDA ($b=b_3$)	SVM
Banana	1	11.9%	11.6%	11.8%	11.4%
B.Cancer	1	35.1%	51.9%	28.6%	26.0%
Diabetes	5	24.3%	26.7%	23.0%	23.0%
F.Sonar	5	33.8%	34.3%	34.5%	34.3%
German	5	23.0%	28.3%	20.0%	21.0%
Heart	5	19.0%	19.0%	18.0%	19.0%
Image	1	4.2%	4.8%	3.8%	2.2%
Ringnorm	5	3.7%	4.0%	3.2%	2.4%
Splice	5	10.3%	10.3%	10.3%	9.8%
Thyroid	1	4.0%	2.7%	2.7%	2.7%
Titanic	5	25.8%	25.8%	22.9%	22.9%
Twonorm	5	2.9%	3.1%	3.8%	3.8%
Waveform	5	10.7%	12.7%	10.5%	10.6%

V. CONCLUSION

In this paper, we proposed a fast and stable iterative algorithm for the kernel linear discriminant analysis in two-class case to avoid the general eigenvalue decomposition

problem in singular case. The binary classifier KLDA is the nonlinear form of linear fisher discriminant classifier, which is an important technique in the statistical pattern recognition. The result of the iterative algorithm can be controlled not to overfit the data (for example, Fig.1 (c)). Experimental results show that if the offset b in (10) is well selected, the performance of KLDA is competitive with SVM.

The separate hyperplane found by KLDA in the feature space is related to all training samples, and this increases the computational complexity in the testing procedure. We can use some reduced set (similar to support vectors) to approximate the hyperplane [8], [9], and testing procedure will be expedited evidently.

ACKNOWLEDGMENT

The authors would like to thank A.M. Martinez and R. Benavente for providing the AR Face Database, and thank G. Rätsch, T. Onoda, etc, for providing the Benchmark Repository.

REFERENCES

[1] B. Scholköpf, A. Smola, and K.R. Müller, "Nonlinear Component Analysis as a Kernel Eigenvalue Problem," Neural Computation, vol.10, pp. 1299–1319, 1998.
[2] G. Baudat and F. Anouar, "Generalized Discriminant Analysis Using a Kernel Approach," Neural Computation, vol.12, pp. 2385–2404, 2000.
[3] C. Park and H. Park, "Fingerprint Classification Using Nonlinear Discriminant Analysis," *Technical Report*, *TR 03-034*, University of Minnesota, USA, Sep. 2003.
[4] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.R. Müller, "Fisher discriminant analysis with kernels," *Neural Networks for Signal Processing IX*, pp. 41–48, 1999.
[5] P.N. Belhumeour, J.P. Hespanha, and D.J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection," *IEEE Trans. Pattern Analysis and Machine Intell.*, vol.19, pp. 711–720, 1997.
[6] H.C. Kim, D. Kim, and S.Y. Bang, "Face recognition using LDA mixture model," *Pattern Recognition Letters*, vol.24, pp. 2815–2821, 2003.
[7] G. Rätsch, T. Onoda, and K.R. Müller, "Soft Margins for AdaBoost," *Machine Learning*, pp. 1–35, 2000.
[8] C. Burges, "Simplified Support Vector Decision Rules," *in Proceedings of the 13th International Conference on Machine Learning*, pp. 71–77, 1996.
[9] B. Schölkopf, P. Knirsch, A. Smola, and C. Burges, "Fast Approximation of Support Vector Kernel Expansions, and an Interpretation of Clustering as Approximation in Feature Spaces," *Proceedings of the DAGM Symposium Mustererkennung*, pp. 124–132, 1998.