A FAST KERNEL DIMENSION REDUCTION ALGORITHM WITH APPLICATIONS TO FACE RECOGNITION

SENJIAN AN, WANQUAN LIU, SVETHA VENKATESH, RONNY TJAHYADI

Department of Computing, Curtin University of Technology, GPO Box U1987 Perth WA 6845
Email: senjian, wanquan, svetha, tjahyadi@cs.curtin.edu.au

Abstract:

This paper presents a novel dimensionality reduction algorithm for kernel based classification. In the feature space, the proposed algorithm maximizes the ratio of the squared between-class distance and the sum of the within-class variances of the training samples for a given reduced dimension. This algorithm has lower complexity than the recently reported kernel dimension reduction(KDR) for supervised learning. We conducted several simulations with large training datasets, which demonstrate that the proposed algorithm has similar performance or is marginally better compared with KDR whilst having the advantage of computational efficiency. Further, we applied the proposed dimension reduction algorithm to face recognition in which the number of training samples is very small. This proposed face recognition approach based on the new algorithm outperforms the eigenface approach based on the principle component analysis (PCA), when the training data is complete, that is, representative of the whole dataset.

Key Words

Support Vector Machine, Dimensional Reduction, Classification, Face Recognition, Optimization.

1 Introduction

Dimension reduction is an essential and powerful technique for many applications since it reduces noise, irrelevant variables and computation complexity. By proper dimension reduction, the classification performance can be improved via removal of noise vectors and irrelevant variables. For the task of removing noise and irrelevant variables, linear dimensionality reduction is usually preferable compared with nonlinear dimensionality reduction due to its simplicity. However, many popular classification methods such as support vector machines (SVM) [2, 12] formulate the classifier in the feature space into which the input data is mapped by nonlinear mapping.

Let $\{(x_i, y_i), i = 1, 2, \cdots, n\}$ be a set of training samples, where the ith example $x_i \in \mathbb{R}^m$ in a m-dimension input space belongs to one of the two classes labeled by $y_i \in \{1, -1\}$. The goal of the SVM is to define a hyperplane in a high-dimensional feature space, which divides the set of samples in the feature space such that all the points with the same label are on the same side of the hyperplane. The mapping from the input space to the feature space is usually nonlinear. One important property of SVM [2, 12] is that it finds an optimal separating hyperplane so as to separate two classes of patterns with maximal margin. The generalization ability of SVMs is related to the margin with which it separates the classes. A modified version of SVM, the least squares support vector machine (LS-SVM) was proposed by [9]. The major difference of LS-SVM with SVM is that an L_2 norm is taken with equality constraints so as to obtain a linear set of equations instead of a quadratic programming problem which is involved in formulating SVM.

[3] proposed kernel dimensionality reduction (KDR), an algorithm for regression or classification problems. They treat the dimensionality reduction problem as that of finding a low-dimensional effective subspace for the explanatory variable x which retains the statistical relationship between x and the observation y, where y can be discrete or continuous. With a general nonparametric characterization of conditional independence via using the variable covariance on reproducing kernel Hilbert spaces, they proposed a contrast function for estimation of the effective subspaces. One advantage of KDR is that it requires neither assumptions on the marginal distribution of x, nor a parametric model of the conditional distribution of the observation. However, the contrast function of KDR involves the inverse of the kernel matrix and the computation complexity of this matrix inversion is generally of order $O(n^3)$ where n is the number of training samples.

Our motivation lies in designing classifiers in feature space into which we have mapped the input using nonlinear kernel functions. To enable such classifiers, a desirable feature is that the classes mapped into the feature space

0-7803-9091-1/05/$20.00 ©2005 IEEE

should be as separable as possible with a linear separating hyperplane. One of the measures of such separability is the ratio of the squared between-class distance and the sum of the within-class variances of the classes mapped into this feature space and we choose it as a contrast function. The proposed algorithm maximizes this contrast function for a given dimension. The significance lies in reducing the complexity of dimensionality reduction to $O(n^2)$ compared to $O(n^3)$ of KDR, whist maintaining comparable classification performance. The motivation of the dimensional reduction in this paper is to improve the classification performance for any SVM-based classifier. Maximization of the proposed criterion is equivalent to feature discriminant analysis (FDA) if the kernel induced mapping from the input space to the feature space is linear.

The layout of this paper is as follows. In Section 2, we briefly review the formulation of LS-SVM. In Section 3, we develop the dimensionality reduction method for clustered data by first presenting the contrast function and then addressing the algorithmic issues. In Section 4, we provide experimental examples to illustrate the performance of the proposed algorithm with a comparison to KDR. In addition, we apply the technique to face recognition and compare the results with the eigenface approach [11].

2 Least Squares Support Vector Machines

Given a training set $\{(x_i, y_i)\}_{i=1}^n$ with input data $x_i \in \mathbb{R}^m$ and class labels $y_i \in \{-1, 1\}$, according to [9, 8], the least square support vector machine (LS-SVM) is formulated as follows

$$\min_{w,b,e} \mathcal{J}(w, e) = \frac{1}{2} w^T w + \gamma \frac{1}{2} \sum_{i=1}^{n} e_i^2 \tag{1}$$

with constraints

$$y_i[w^T \varphi(x_i) + b] = 1 - e_i, i = 1, 2, \cdots, n. \tag{2}$$

The nonlinear function $\varphi(\cdot) : \mathbb{R}^m \to \mathbb{R}^n$, which is usually induced by a kernel function, maps the input space to a high dimensional feature space. The classifier in primal space is formulated as

$$y(x) = \text{sign}[w^T \varphi(x) + b], \tag{3}$$

where w and b are obtained from (1) and (2) and x is the testing data. However, one never need evaluate w and $\varphi(\cdot)$ in the LS-SVM framework. By Lagrangian multiplier optimization methods, the solution of the minimization problem in the primal space can be obtained by solving the following linear system (see [8] for details)

$$\begin{bmatrix} 0 & y^T \\ y & \Omega + I/\gamma \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{1}_n \end{bmatrix} \tag{4}$$

with $y = [y_1, y_2, \cdots, y_n], \mathbf{1}_n = [1, 1, \cdots, 1]^T, \alpha = [\alpha_1, \alpha_2, \cdots, \alpha_n]^T$ and

$$\Omega_{ij} = y_i y_j \varphi(x_i)^T \varphi(x_j) = y_i y_j K(x_i, x_j) \tag{5}$$

where the kernel K satisfies the Mercer's condition. Then, w is of the form

$$w = \sum_{i=1}^{n} \alpha_i y_i \varphi(x_i) \tag{6}$$

and the LS-SVM classifier is constructed as

$$y(x) = \text{sign}\left[\sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b\right]. \tag{7}$$

For the kernel function $K(\cdot, \cdot)$, one typically can choose either linear, polynomial or Gaussian kernels.

Since the evaluation of kernel matrix depends on the dimension of the input space, the dimension reduction of the input space can reduce the computational complexity of the training and formulation of LS-SVM classifiers.

3 Dimension Reduction for Clustered Data

Given a training set $\{(x_i, y_i)\}_{i=1}^n$ with input data $x_i \in \mathbb{R}^m$ and class labels $y_i \in \{-1, 1\}$, the task of dimension reduction for kernel based classification is to find a matrix $B \in \mathbb{R}^{m \times r}$ $(r < m)$ with $B^T B = I_r$ such that the set $\{z_i = B^T x_i\}_{i=1}^n$ is separable with the largest possible margin by a hyperplane in the feature space. In this paper, we choose the ratio of the squared between-class distance and the sum of the within class variances of the training samples in the feature space as a criterion to be maximized under the constraint $B^T B = I_r$. The constraint $B^T B = I_r$ is necessary in signal processing as demonstrated in [11]. Therefore,

$$F(B) = \frac{d^2}{v_1^2 + v_2^2} \tag{8}$$

where d denotes the between-class distance which is defined as the distance of the centers of the two different classes, i.e,

$$d \doteq \left\| \frac{1}{n_1} \sum_{i=1}^{n_1} \varphi(z_i) - \frac{1}{n_2} \sum_{i=n_1+1}^{n} \varphi(z_i) \right\|, \tag{9}$$

and v_1^2, v_2^2 denote the within-class variances

$$\begin{aligned} v_1^2 &= \tfrac{1}{n_1} \textstyle\sum_{i=1}^{n_1} \left\| \varphi(z_i) - \tfrac{1}{n_1} \sum_{i=1}^{n_1} \varphi(z_i) \right\|^2 \\ v_2^2 &= \tfrac{1}{n_2} \textstyle\sum_{i=n_1+1}^{n} \left\| \varphi(z_i) - \tfrac{1}{n_2} \sum_{i=n_1+1}^{n} \varphi(z_i) \right\|^2. \end{aligned} \tag{10}$$

Here, n_1 is the number of samples in cluster one with labels 1, n_2 is the number of samples in cluster two with labels -1 and $n_1 + n_2 = n$. Also, without loss of generality, the samples $\{x_i\}_{i=1}^n$ are organized such that the first n_1 samples are the samples in cluster one and the remaining are the samples in cluster two.

Throughout this paper, we use the Gaussian kernel

$$k(z_i, z_j) = e^{-\frac{\|z_i - z_j\|^2}{\sigma^2}}$$

where $z_i = B^T x_i$. The Gram matrix of z is defined as $(G)_{ij} = k(z_i, z_j)$. The Kernel matrix K is defined as

$$K = WGW \tag{11}$$

where $W = I_n - \mathbf{1}_n \mathbf{1}_n^T$ and $\mathbf{1}_n$ denotes a vector with all elements being 1. Using the fact that $K_{ij} = \langle \varphi(z_i), \varphi(z_j) \rangle = \varphi(z_i)^T \varphi(z_j)$, by direct calculation, we obtain

$$\begin{aligned} d^2 &= \tfrac{1}{n_1^2} \textstyle\sum_{i=1}^{n_1} \sum_{j=1}^{n_1} K_{ij} \\ &\quad + \tfrac{1}{n_2^2} \textstyle\sum_{i=n_1+1}^{n} \sum_{j=n_1+1}^{n} K_{ij} \\ &\quad - \tfrac{2}{n_1 n_2} \textstyle\sum_{i=1}^{n_1} \sum_{j=n_1+1}^{n} K_{ij} \\ &= \begin{bmatrix} \frac{1}{n_1}\mathbf{1}_{n_1}^T & -\frac{1}{n_2}\mathbf{1}_{n_2}^T \end{bmatrix} K \begin{bmatrix} \frac{1}{n_1}\mathbf{1}_{n_1} \\ -\frac{1}{n_2}\mathbf{1}_{n_2} \end{bmatrix} \end{aligned} \tag{12}$$

Note that

$$W \begin{bmatrix} \frac{1}{n_1}\mathbf{1}_{n_1} \\ -\frac{1}{n_2}\mathbf{1}_{n_2} \end{bmatrix} = \begin{bmatrix} \frac{1}{n_1}\mathbf{1}_{n_1} \\ -\frac{1}{n_2}\mathbf{1}_{n_2} \end{bmatrix}. \tag{13}$$

Replacing K_{ij} by G_{ij}, (12) is still true, i.e.,

$$\begin{aligned} d^2 &= \begin{bmatrix} \frac{1}{n_1}\mathbf{1}_{n_1}^T & -\frac{1}{n_2}\mathbf{1}_{n_2}^T \end{bmatrix} K \begin{bmatrix} \frac{1}{n_1}\mathbf{1}_{n_1} \\ -\frac{1}{n_2}\mathbf{1}_{n_2} \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{n_1}\mathbf{1}_{n_1}^T & -\frac{1}{n_2}\mathbf{1}_{n_2}^T \end{bmatrix} WGW \begin{bmatrix} \frac{1}{n_1}\mathbf{1}_{n_1} \\ -\frac{1}{n_2}\mathbf{1}_{n_2} \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{n_1}\mathbf{1}_{n_1}^T & -\frac{1}{n_2}\mathbf{1}_{n_2}^T \end{bmatrix} G \begin{bmatrix} \frac{1}{n_1}\mathbf{1}_{n_1} \\ -\frac{1}{n_2}\mathbf{1}_{n_2} \end{bmatrix} \\ &= \tfrac{1}{n_1^2} \textstyle\sum_{i=1}^{n_1} \sum_{j=1}^{n_1} G_{ij} \\ &\quad + \tfrac{1}{n_2^2} \textstyle\sum_{i=n_1+1}^{n} \sum_{j=n_1+1}^{n} G_{ij} \\ &\quad - \tfrac{2}{n_1 n_2} \textstyle\sum_{i=1}^{n_1} \sum_{j=n_1+1}^{n} G_{ij} \end{aligned} \tag{14}$$

If we denote

$$\begin{aligned} \alpha(B) &= \tfrac{1}{n_1^2} \textstyle\sum_{i=1}^{n_1} \sum_{j=1}^{n_1} G_{ij} \\ &\quad + \tfrac{1}{n_2^2} \textstyle\sum_{i=n_1+1}^{n} \sum_{j=n_1+1}^{n} G_{ij} \\ \beta(B) &= \tfrac{2}{n_1 n_2} \textstyle\sum_{i=1}^{n_1} \sum_{j=n_1+1}^{n} G_{ij}, \end{aligned} \tag{15}$$

then

$$d^2 = \alpha(B) - \beta(B). \tag{16}$$

Similarly, one has

$$\begin{aligned} v_1^2 &= \tfrac{1}{n_1} \textstyle\sum_{i=1}^{n_1} K_{ii} - \tfrac{1}{n_1^2} \sum_{i=1}^{n_1} \sum_{j=1}^{n_1} K_{ij} \\ v_2^2 &= \tfrac{1}{n_2} \textstyle\sum_{i=n_1+1}^{n} K_{ii} - \tfrac{1}{n_2^2} \sum_{i=n_1+1}^{n} \sum_{j=n_1+1}^{n} K_{ij} \end{aligned} \tag{17}$$

Let $a = [a_1, a_2, \cdots, a_n]^T \doteq G\mathbf{1}_n$ and $\Delta \doteq K - G$. Then $\Delta_{ij} = -\frac{1}{n}(a_i + a_j) + \frac{1}{n^2} \sum_{i=1}^n a_i$. It is straightforward to verify that

$$\begin{aligned} \tfrac{1}{n_1} \textstyle\sum_{i=1}^{n_1} \Delta_{ii} - \tfrac{1}{n_1^2} \sum_{i=1}^{n_1} \sum_{j=1}^{n_1} \Delta_{ij} = 0 \\ \tfrac{1}{n_2} \textstyle\sum_{i=n_1+1}^{n} \Delta_{ii} - \tfrac{1}{n_2^2} \sum_{i=n_1+1}^{n} \sum_{j=n_1+1}^{n} \Delta_{ij} = 0 \end{aligned} \tag{18}$$

Hence, replacing K_{ij} by G_{ij}, (17) and (8) are still true, and therefore $v_1^2 + v_2^2 = 2 - \alpha(B)$ which implies that

$$F(B) = -1 + \frac{2 - \beta(B)}{2 - \alpha(B)} \tag{19}$$

Therefore, the aim of this paper is to maximize the contrast function $F(B)$ with unitary constraint $B^T B = I$.

3.1 The Algorithm

The maximization problem of $F(B)$ is a special type of optimization problem under unitary constraints and we apply the optimization algorithm proposed in [6] to solve it. The only prerequisite for being able to implement this algorithm is to compute the derivative of the cost function. Since our cost function satisfies $F(B) = F(BQ)$ for any unitary matrix Q, it should be maximized on the Grassmann manifold. In this paper, we adopt the Algorithm 24 [6] which can be summarized as follows.

1. Choose $B \in R^{m \times r}$ such that $B^T B = I$. Set step size $\gamma := 1$.
2. Compute D_B, which is the derivative of F at B.
3. Compute the descent direction $Z := (I - BB^T)D_B$.
4. Evaluate $\langle Z, Z \rangle = tr\{Z^T Z\}$. If $\sqrt{\langle Z, Z \rangle}$ is sufficiently small, then stop.
5. If $F(\pi\{B + 2\gamma Z\}) - F(B) > \gamma\langle Z, Z \rangle$, then set $\gamma := 2\gamma$ and repeat 5.
6. If $F(\pi\{B + \gamma Z\}) - F(B) < 0.5\gamma\langle Z, Z \rangle$, then set $\gamma := 0.5\gamma$ and repeat 6.
7. Set $B := \pi\{B + \gamma Z\}$. Go to Step 2.

Here $\pi\{B\}$ denotes an orthogonal basis of $span(B)$. The threshold in step 4 for the stopping criterion depends on the properties of the optimal point of the cost function, i.e., the flatness of the cost function around the optimal point. However, it is usually good enough to choose the threshold as 0.001. The initial value for this algorithm is chosen as below in this paper. With training samples $\{x_i\}_{i=1}^n$ and $x_i \in R^m$, one can construct a matrix

$$X = [x_1, x_2, \cdots x_n]$$

and its covariance matrix XX^T. The eigenvectors corresponding to the largest r eigenvalues will be chosen as the initial condition B_0. This selection of initial condition is actually based on PCA technique [5].

Next, we derive the formula for the derivative of the cost functions $F(B)$. Note that

$$G_{ij} = e^{-\frac{\|B^T x_i - B^T x_j\|^2}{\sigma^2}}$$

and

$$\frac{\partial G_{ij}}{\partial B} = \frac{-2G_{ij}}{\sigma^2}(x_i - x_j)(x_i - x_j)^T B. \tag{20}$$

The derivatives of $\alpha(B)$ and $\beta(B)$ are:

$$\begin{aligned} \tfrac{\partial \alpha}{\partial B} &= \tfrac{-2}{\sigma^2}\left\{ \tfrac{1}{n_1^2}\textstyle\sum_{i=1}^{n_1}\sum_{j=1}^{n_1} G_{ij}\delta x_{ij}\delta x_{ij}^T \right. \\ &\quad \left. + \tfrac{1}{n_2^2}\textstyle\sum_{i=n_1+1}^{n}\sum_{j=n_1+1}^{n} G_{ij}\delta x_{ij}\delta x_{ij}^T \right\} B \qquad (21) \\ \tfrac{\partial \beta}{\partial B} &= \tfrac{-2}{\sigma^2}\left\{ \tfrac{2}{n_1 n_2}\textstyle\sum_{i=1}^{n_1}\sum_{j=n_1+1}^{n} G_{ij}\delta x_{ij}\delta x_{ij}^T \right\} B \end{aligned}$$

where $\delta x_{ij} = x_i - x_j$, and thus the derivative of $F(B)$ can be obtained as

$$\frac{\partial F}{\partial B} = \frac{2(\alpha' - \beta') + \alpha\beta' - \alpha'\beta}{(2-\alpha)^2} \tag{22}$$

By simplification, one can find P such that,

$$\frac{\partial F}{\partial B} = XPX^T B \tag{23}$$

where $X = [x_1, x_2, \cdots, x_n] \in R^{m \times n}$ and P can be obtained as follows. Decompose G into four blocks

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{12}^T & G_{22} \end{bmatrix} \tag{24}$$

where $G_{11} \in \mathbb{R}^{n_1 \times n_1}, G_{22} \in \mathbb{R}^{n_2 \times n_2}$ and $G_{12} \in \mathbb{R}^{n_1 \times n_2}$.

Let

$$\begin{aligned} \bar{G} &= \begin{bmatrix} \frac{2-\beta}{n_1^2} G_{11} & \frac{2-\alpha}{n_1 n_2} G_{12} \\ \frac{2-\alpha}{n_1 n_2} G_{12}^T & \frac{2-\beta}{n_1^2} G_{22} \end{bmatrix} \qquad (25) \\ \bar{D} &= diag\{\bar{d}_1, \bar{d}_2, \cdots, \bar{d}_n\} \end{aligned}$$

where $\bar{d}_i$ is the sum of the ith row of $\bar{G}$. Then

$$P = \frac{4}{\sigma^2(2-\alpha)^2}(\bar{G} - \bar{D}). \tag{26}$$

This completes the development of the dimension reduction algorithm. Once we can find an optimal B, we can develop the classification approach in lower feature space based on the LS-SVM. Since the proposed reduction algorithm is a SVM-Oriented Dimension Reduction (SDR) approach, we will denote it as SDR in the rest part of this paper.

3.2 Computation Complexity

In order to estimate the complexity of the proposed algorithm, we first consider the evaluation of the Gram matrix. It can be implemented as follows.

1. Compute $z_i = B^T x_i$ for $i = 1, 2, \cdots, n$;
2. Let
$$Z = [z_1, z_2, \cdots, z_n] \in R^{r \times n}$$
and compute $M = Z^T Z$;
3. Let a denote the diagonal vector of M,i.e.,$a(i) = M_{ii}$, and compute
$$Q = a\mathbf{1}_n^T + \mathbf{1}_n a^T - 2M$$
Note that $Q_{ij} = \|z_i - z_j\|^2$.
4. Evaluate G with $G_{ij} = exp(-Q_{ij}/\sigma^2)$.

For each step, the complexity order is less than or equal to $O(n^2)$. Note that we assume that $m \ll n$. So the evaluation of the Gram matrix is of order $O(n^2)$. Once G obtained, from equations (15,19,23,25,26), one can see that the evaluation of the cost function and its derivative is also of the order $O(n^2)$. Hence, the computation complexity of the overall algorithm is of order $O(n^2)$. Since the cost function of KDR involves the inverse of the kernel matrix with dimension $n \times n$ and matrix inversion is generally an $O(n^3)$ process[1], the proposed algorithm has lower complexity than KDR.

4 Experimental Results

4.1 Application to Two Benchmark Datasets

In this section, we first report the application of the proposed ratio maximization algorithm and compare it with KDR on the two benchmark datasets from UCI benchmark repository [1]: a separable one, the Johns Hopkins university ionosphere(ion), and a noisy one, the Statlog heart disease(hea). The hea dataset consists of 270 samples with dimension 13 while the ion dataset consists of 351 samples with dimension 33. The experiments have been carried out for 100 randomizations, for each randomization 2/3 of the data is chosen for dimension reduction and for training of LS-SVM classifiers and the remaining 1/3 is used for testing. The LS-SVM algorithm is downloaded from *http://www.esat.kuleuven.ac.be/sista/lssvmlab/*. Theoretically, we should train the hyper-parameters, i.e., the

[1] Theoretically, the complexity of matrix inversion can be reduced to $O(n^{2.496})$[7]

kernel function parameter σ (used in dimension reduction and SVM classification) and the regularization constant γ (used in the SVM classification) for each dimension-reduced dataset and therefore the optimal dimension reduction matrix B. Thus, the optimal hyper-parameters should be optimized alternately. For simplicity, in the training of LS-SVM classifiers, we choose the optimal hypterparameters as suggested in [4] regardless of the dimension reduction matrix B. The optimal chosen parameters for σ and γ are: $\sigma = 5.69, \log_{10}(\gamma) = -0.76$ for the hea dataset, and $\sigma = 3.30, \log_{10}(\gamma) = 0.63$ for the ion dataset.

Figure 1 shows the average classification rates of 100 randomizations for various dimensions. The figure shows the classification rates of LS-SVMs for three cases: a), using all variables (no dimension reduction), b), using the proposed ratio maximization to reduce the dimension and c), using KDR to reduce the dimension. Compared to using all variables, both ratio maximization and KDR maintain a comparable performance after dimension reduction while ratio maximization method performs better for the heart disease dataset and KDR performs slightly better for the ionosphere dataset.

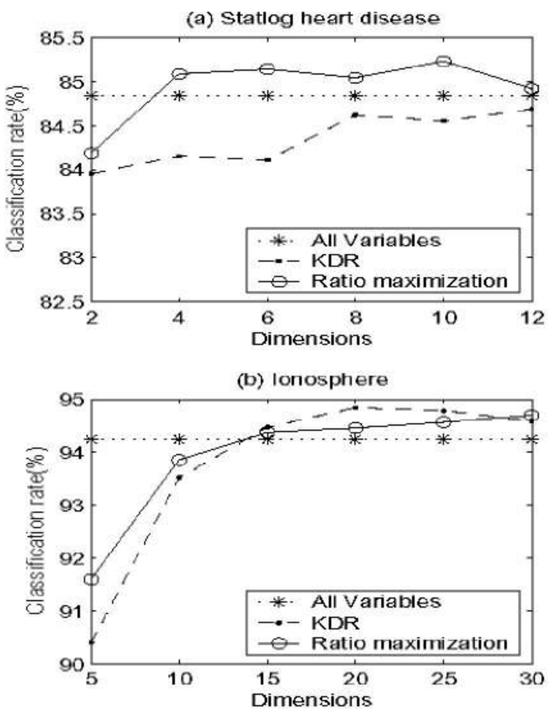

Figure 1. Classification accuracy of the LS-SVM after dimension reduction.

Table 1. Running times (s) for KDR on hea data set with various training samples and dimensions.

Samples/dims	4	8	12
90	14.2810	20.6250	26.9220
180	83.2810	110.4840	140.3600
270	259.8280	337.0940	419.5150

Table 2. Running times (s) for the proposed algorithm on hea data set with various training samples and dimensions.

Samples/dims	4	8	12
90	1.7180	1.9380	2.0000
180	6.4530	7.1870	7.7340
270	15.2500	16.6720	17.3900

Table 1 and Table 2 shows the time required to conduct KDR and the proposed algorithm with various samples and dimensions on the hea dataset. KDR is much more time-consuming than the proposed algorithm as expected. Table 1 and Table 2 can approximately verify that the complexity of the proposed algorithm and KDR is of order $O(n^2)$ and $O(n^3)$ respectively.

4.2 Application to Face Recognition

In above experiment, the number of training samples is sufficiently large and there are only two classes for all the samples. Next, we will apply the proposed SDR algorithm to face recognition and investigate its effectiveness. Compared to the previous example, the number of training samples in this experiment is small and further there are more than two classes for all the experiments.

Experiments were carried out on ten datasets created from Yale database [13]. This database contains 15 individuals (mostly male) with 11 images each. Table 3 shows some of the images used in the training and testing datasets. Each training dataset was constructed from 15 individuals with 4 images each. The remaining images not included in training dataset are used to construct the corresponding testing datasets. No preprocessing methods are used to enhance the facial images prior to feature extraction. In order to investigate the SDR fairly, face images with light configurations as shown in Figure 2 were excluded as the excessive light casts shadows on the background which requires preprocessing in practice.

Also the eigenface algorithm based on PCA from [11] is

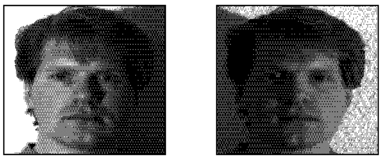

Figure 2. Images with light configurations from Yale face database.

Dataset		Images
1	Training	happy, normal, sleepy, wink
	Testing	glasses, no-glasses, sad, surprised
2	Training	glasses, happy, no-glasses, normal
	Testing	sad, sleepy, surprised, wink
3	Training	glasses, happy, surprised, wink
	Testing	no-glasses, normal, sad, sleepy
4	Training	happy, no-glasses, normal, sad
	Testing	glasses, sleepy, surprised, wink
5	Training	glasses, no-glasses, sleepy, surprised
	Testing	happy, normal, sad, wink
6	Training	glasses, normal, sad, surprised
	Testing	happy, no-glasses, sleepy, wink
7	Training	happy, no-glasses, sad, surprised
	Testing	glasses, normal, sleepy, wink
8	Training	no-glasses, sad, sleepy, wink
	Testing	glasses, happy, normal, surprised
9	Training	glasses, no-glasses, normal, sleepy
	Testing	happy, sad, surprised, wink
10	Training	no-glasses, normal, sad, sleepy
	Testing	glasses, happy, surprised, wink

Table 3. Images used in Yale training and testing datasets.

implemented for comparison with the proposed face recognition approach based on the SDR since the eigenface approach has been broadly used [10]. The face recognition system in Figure 3 shows the implementation of the SDR with LS-SVM classifier. This system consists of 2 stages, namely training and recognition. Since the sample data size M is much smaller than its dimension and the rank of the data matrix is at most M, we use the PCA [5] to remove its null vectors and thus reduce the training sample to dimension M. This creates eigenspace $E1$ with the number (M') of eigenvectors being set to be the number of training images (M). Then the training images are projected into the eigenspace $E1$ and these projections are used as training data (Xtr) for SDR reduction algorithm. The next step is to find the optimal parameters for σ and γ using the *tunelssvm* function downloaded from *http://www.esat.kuleuven.ac.be/sista/lssvmlab/*. With Xtr, we use the SDR dimension reduction algorithm to obtain a lower-order training sample Ztr based on which the LS-SVM classifiers are trained. We denote the dimension of Ztr as r. Each LS-SVM model represents an individual and it is trained to output '1' on the corresponding individual and output '0' on other individuals. In the recognition stage, the test image is first projected into the eigenspace $E1$ and then further reduced by SDR producing $Ztest$. We then verify which individual $Ztest$ belongs to via each LS-SVM model with the following classifier:

$$y(x) = \sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b. \tag{27}$$

If the test stage produces *only* one positive signal, then the testing image matches the corresponding individual. Otherwise, we iteratively train new LS-SVM models taking data for individuals corresponding to positive signals since these individuals are possible right classes according to the LS-SVM. This process is repeated with each test image until *only* one positive signal is produced. Thus, the testing process continues to produce a hierachy of LS-SVM's.

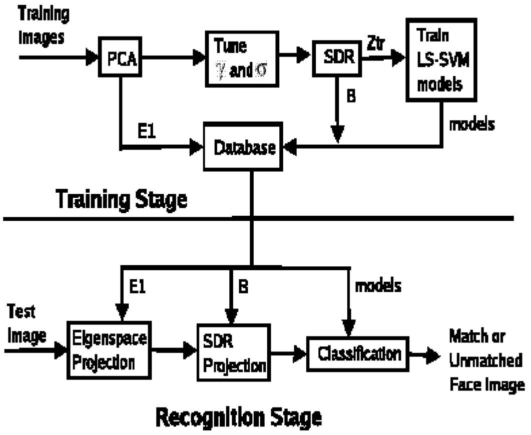

Figure 3. Face Recognition with SDR and LS-SVM.

The simulation results are shown in Table 4. Figure 4 gives the average performance for these two approaches. One can see that from dimension 20 to 22, the SDR performs better on average than the eigenface approach. In detail, one can see from Table 4 that the SDR only performs worse than the eigenface approach in testing datasets 9 and 10, while performing much better in other cases in these dimension ranges. One can see that the training samples in dataset 10 do not cover the case of wearing glasses while it appears in

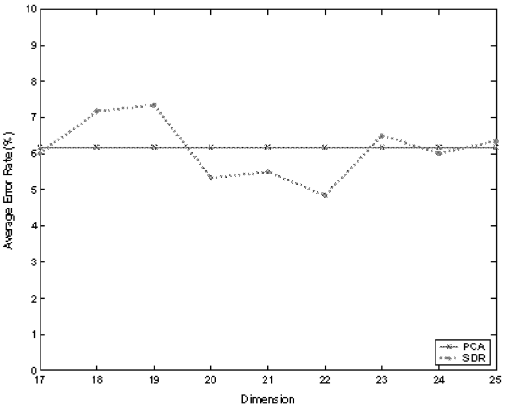

Figure 4. Average Error Rates of SDR vs PCA.

the testing samples. Similar cases happen in dataset 9 in which the face expressions in the training samples are not sufficiently complete to cover all the possible cases in the testing samples. Based on these observations, we may conclude that the SDR should perform better that the eigenface approach based on PCA if the training samples are complete to cover the salient features of the dataset. SInce SDR aims to maximize the separability of the training samples, it requires that the training samples are complete enough so that their distribution are approximately the same as that of the dataset including test samples.

4.3 Face Recognition via LS-SVM

Since the proposed contrast function in this paper is motivated by SVM-based classification idea, in this section, we will conduct some experiments on face recognition with the proposed SDR reduction algorithm, and without any dimension reduction. As demonstrated in the previous section, the SVM-based classification approaches require the training samples to be as complete as possible. As seen in Table 3, datasets 5 and 6 satisfy this essential requirement.

In detail, we will use the Xtr as the training sample with LS-SVM as a classifier and do experiments on datasets 5 and 6. Further, we also use the Ztr as the training sample with LS-SVM as classifier and do the experiments on these two datasets. Their error rates are displayed in Figure 5 and Figure 6 respectively. These figures showed that the performance is much better after SDR dimensional reduction. This illustrates that dimension reduction is necessary and can indeed improve the performance.

Dataset/ Dimension		17	18	19	20	21
1	SDR	1.7%	5.0%	6.7%	3.3%	1.7%
	PCA	8.3%	8.3%	8.3%	8.3%	8.3%
2	SDR	8.3%	1.7%	3.3%	3.3%	5.0%
	PCA	5.0%	5.0%	5.0%	5.0%	5.0%
3	SDR	3.3%	1.7%	5.0%	1.7%	0.0%
	PCA	10.0%	10.0%	10.0%	10.0%	10.0%
4	SDR	10.0%	10.0%	11.7%	5.0%	5.0%
	PCA	6.7%	6.7%	6.7%	6.7%	6.7%
5	SDR	0.0%	1.7%	1.7%	1.7%	1.7%
	PCA	1.7%	1.7%	1.7%	1.7%	1.7%
6	SDR	0.0%	0.0%	3.3%	3.3%	1.7%
	PCA	5.0%	5.0%	5.0%	5.0%	5.0%
7	SDR	5.0%	6.7%	6.7%	5.0%	5.0%
	PCA	5.0%	5.0%	5.0%	5.0%	5.0%
8	SDR	8.3%	10.0%	5.0%	5.0%	5.0%
	PCA	10.0%	10.0%	10.0%	10.0%	10.0%
9	SDR	8.3%	15.0%	13.3%	11.7%	10.0%
	PCA	3.3%	3.3%	3.3%	3.3%	3.3%
10	SDR	15.0%	20.0%	16.7%	13.3%	20.0%
	PCA	6.7%	6.7%	6.7%	6.7%	6.7%

Dataset/ Dimension		22	23	24	25
1	SDR	3.3%	5.0%	3.3%	3.3%
	PCA	8.3%	8.3%	8.3%	8.3%
2	SDR	6.7%	6.7%	6.7%	8.3%
	PCA	5.0%	5.0%	5.0%	5.0%
3	SDR	0.0%	0.0%	1.7%	0.0%
	PCA	10.0%	10.0%	10.0%	10.0%
4	SDR	3.3%	6.7%	1.7%	8.3%
	PCA	6.7%	6.7%	6.7%	6.7%
5	SDR	1.7%	5.0%	6.7%	5.0%
	PCA	1.7%	1.7%	1.7%	1.7%
6	SDR	0.0%	1.7%	1.7%	1.7%
	PCA	5.0%	5.0%	5.0%	5.0%
7	SDR	5.0%	5.0%	6.7%	5.0%
	PCA	5.0%	5.0%	5.0%	5.0%
8	SDR	6.7%	6.7%	6.7%	6.7%
	PCA	10.0%	10.0%	10.0%	10.0%
9	SDR	6.7%	8.3%	8.3%	8.3%
	PCA	3.3%	3.3%	3.3%	3.3%
10	SDR	15.0%	20.0%	16.7%	16.7%
	PCA	6.7%	6.7%	6.7%	6.7%

Table 4. SDR and PCA error rates on ten datasets.

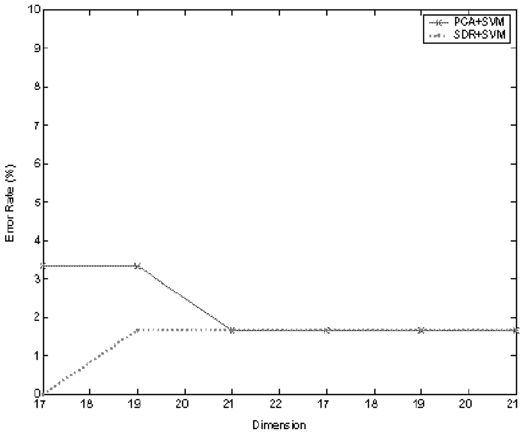

Figure 5. Error Rates of SDR vs PCA

5 Conclusions

In this paper, a novel, linear dimension reduction algorithm for clustered data is developed by maximizing the ratio of the squared between-class distance and the sum of the within-class variances of training samples mapped into

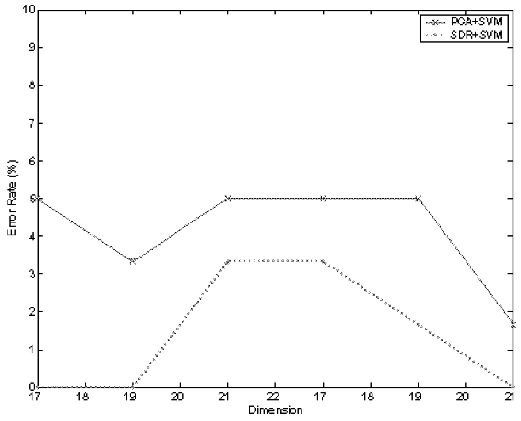

Figure 6. Error Rates of SDR vs PCA

the feature space. After dimension reduction, the computation complexity of the formulating of the LS-SVM is then reduced. This algorithm has an advantage of computational efficiency while maintaining comparable performance compared to KDR for supervised learning. Further, we showed that the proposed approach can also applied to face recognition in which the number of training samples is small. The experiments show that dimensional reduction is necessary in face recognition and that the proposed technique can indeed improve the performance.

References

[1] C. L. Blake and C. J. Merz. *UCI repository of machine learning databases [http://www.ics.uci.edu/ mlearn/MLRepository.html].* Irvine, CA: University of California, Dept. of Information and Computer Science, 1998.

[2] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proc. Of the Fifth Annual Workshop on Computational Learning Theory*, pages 144–152, 1992.

[3] K. Fukumizu, F. R. Bach, and M. I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

[4] T. V. Gestel, J. A. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. D. Moor, and J. Vandewalle. Benchmarking least squares support vector machine classifiers. *machine learning*, 54:5–32, 2004.

[5] G. H. Golub and C. F. V. Loan. *Matrix Computations, 2nd ed.* Baltimore: Johns Hopkins University Press, 1989.

[6] J. H. Manton. Optimization algorithms exploiting unitary constraints. *IEEE Trans. Signal Processing*, 50:635–650, 2002.

[7] V. Pan. How can we speed up matrix multiplication? *SIAM Review*, 26:393–415, 1984.

[8] J. Suykens, T. V. Gestel, J. Brabanter, B. D. Moor, and J. Vandewalle. *Least squares support vector machines.* World Scientific, 2002.

[9] J. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9:293–300, 1999.

[10] R. Tiahyadi. *Investigations into PCA and DCT Based Recognition Algorithms.* Master Thesis, Curtin University of Technology, 2004.

[11] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 13(1):71–86, 1991.

[12] V. Vapnik. *The nature of statistical learning theory.* New-York: Spring-Verlag, 1995.

[13] WebSite. *[Online] http://vismod.media.mit.edu/vismod /classes/mas622-00/datasets.* YALE University Face Database, 2004.