International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A Modified Method for Face Recognition Using SVM

Rong Chen, Yongfeng Cao[1], Hong Sun, Wen Yang

Lab of Signal Processing, School of Electronic Information, Wuhan University ,Wuhan 430079, China

Abstract: Support vector machine is a new technique of machine learning developed from the middle of 1990s. There are two important things in the process of face recognition using SVM. One is the feature vectors extracted from face images, the other is the classifier we choose. Our approach optimizes traditional method from these two points. The adaptively weighted Fisherface purposes on a better feature extraction; the size condensing of training data is used to speed up the modeling of the SVMs; and binary tree structure SVMs is selected here to extend SVM to multi-class cases without much additional complexity. We do our experiments on ORL face database and get good recognition results. The accuracy of recognition results are compared to show the influence of the dimension of the Fisherface. At last, the benefit of adaptively weighted Fisherface is shown by a compare with the traditional Fisherface.

Keywords: Adaptively weighted Fisherface; Center Distance Ratio (CDR); Binary tree structure SVMs; Face recognition

1. Introduction

Face recognition technology has been widely used nowadays such as access control, identity authentication and signature system. Over the past few years, interests and research activities have increased significantly in this field. A robust system for face recognition should be able to deal with the changes in face images[1]. Different images of the same person may vary a lot because of different illumination conditions, poses and details such as glasses. How to solve this problem with a proper complexity becomes a big challenge for researchers.

Researchers have proposed many methods to improve the performance of face recognition, such as Fisherface [2], and adaptively weighted Fisherface [4], or NCC (Nearest Cluster Center) and SVM classifiers [7]. They have been proved effective by experiments. But many modified algorithms are either in the aspect of feature extraction or in the aspect of different classifier design. We regard face recognition as a combination of feature extraction and feature vector classification. So we decide to optimize our approach from both two key points above. On one hand, we use adaptively weighted Fisherface to modify the feature vectors extracted from images, on the other hand, some optimization techniques in data mining, such as data condensing in SVM modeling [8-9], parameter determining by grid search, and binary tree structure SVMs for multi-class classification, are adopted here to make SVMs more efficient in our approach. The whole process of our approach is shown in Figure.1.

From Figure.1 we can see that the features of the training set are extracted first, then we use these feature vectors to train the SVM classifiers. After we get the SVM models, we put the features of the test images into the SVM models to get the classification result.

Our passage is organized as follows: In Section 2, ways to extract features are described, and basic formulations will be given. We'll introduce the basic theory of SVM and introduce techniques of data mining adopted in our approach in Section 3. Experiment results are shown in Section 4 and some conclusions will be given in Section 5.

2. Ways to extract feature from face images

[1] Corresponding author.
Email address:Cao_yong_feng@sohu.com

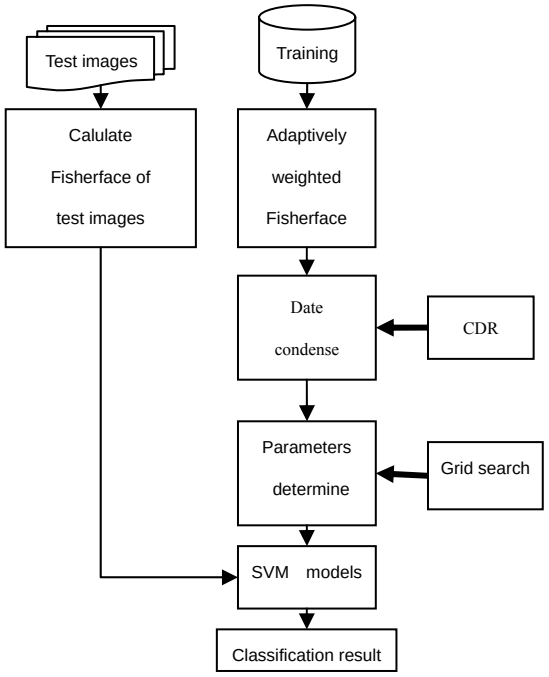


Figure. 1 A complete flow of our modified method for face recognition.

2.1 Traditional approaches of face image processing

Within the last several years, numerous algorithms have been proposed for face recognition. A large number of face recognition techniques use representations found by unsupervised statistical methods. In order to reduce the influence of diversity in poses, some algorithms treat the face as a combination of small parts, such as eyes, nose, mouth, and chin. These are called geometric feature-based methods. They use the properties and relations such as areas, distances, and angles between the small parts as the descriptors of faces. These geometric features-based methods are good at reducing the variety in poses and details. Unfortunately, this class of methods relies heavily on the extraction and measurement of facial features which are hard to achieve up to now.

Besides the geometric feature-based methods, another important class of feature extraction is called global methods. This kind of methods mainly treats each pixel of the image as a coordinate in a high-dimensional space, using the value of each pixel to be the features. As we know, even a small face image may have thousands of pixels, so the dimensions of feature vector may be very large. Many approaches have been proposed to reduce the dimensions of the feature vector of each image without losing much information. The most successful and widely used algorithms are known as PCA (Principle Component Analysis), KLT (Karhunen-Loeve Transform) and the Fisherface. The PCA method maximizes the distances between different classes, but also enlarges the distances within the same class. Compared to PCA, Fisherface algorithm not only enlarges the distance between different classes, but also reduces the distances within class. So we choose Fisherface to be our feature vector and the basic of our optimization.

2.2. Adaptively weighted Fisherface

Fisher linear analysis is based on large database. The center of each class approximately equals to the mean vector of the class, which means the feature vectors of images in certain class locate around the center statistically. But sometimes, the cases we deal with are small databases, which contain a few images for each class. So there exists the possibility that some feature vectors may locate far from the others in feature space. These unexpected points are called outlier. Obviously, these outliers will give a negative influence to the final classification result. Adaptively weighted Fisherface was proposed to solve this problem. At first, we should decrease the dimension of original training data by PCA. The coefficient vector of each image extracted after PCA can be represented as below:

$$y_j^{(i)} = (y_{(j,1)}^{(i)}, y_{(j,2)}^{(i)}, \cdots, y_{(j,m)}^{(i)})^T \quad (1)$$

Where $i = 1,2,\cdots,c$, c denotes to the number of classes; $j = 1,2,\cdots N_i$, N_i denotes to the number of training samples in class i. m is the dimension of the coefficient vector.

We take the 1st dimension of the coefficient vector for example to illustrate how the adaptively weighted Fisherface works. The other dimensions of the vector can be processed in the same way.

Firstly, calculate the sum of distances between each sample in class i and the other samples of the same class: $d_{(j,1)}^{(i)}(j = 1,2,\cdots,N_i)$, then find the minimum of them,

$$d_1^{(i)} = \arg\min_j (d_{(j,1)}^i) \quad (2)$$

We consider that the bigger sum of distance a sample has, the farther away it locates from the center of the class. So we give a small weight to these outliers to reduce the influence they induce. The weight can be determined by the following:

$$\mu_{(j,1)}^{(i)} = 1 + \beta \frac{d_1^{(i)}}{d_{(j,1)}^{(i)}} \quad (3)$$

Where β is a positive constant. It becomes the traditional Fisherface when $\beta = 0$.

The center of class *i* on the 1st dimension can be modified as:

$$\tilde{y}_1^{(i)} = \frac{\sum_{j=1}^{N_i} \mu_{(j,1)}^{(i)} y_{(j,1)}^{(i)}}{\sum_{j=1}^{N_i} \mu_{(j,1)}^{(i)}} \tag{4}$$

This can also be used on the other dimensions of the coefficient vector.

Weighted center of each class:

$$y^{(i)} = (\tilde{y}_1^{(i)}, \tilde{y}_2^{(i)}, \cdots, \tilde{y}_m^{(i)})^T \tag{5}$$

After calculating all the weighted center of each class, the within-class scatter matrix S_w and the between-class scatter matrix S_b can be redefined as:

$$S_w = \frac{1}{N} \sum_{i=1}^{c} \sum_{j=1}^{N_i} (y_j^{(i)} - y^{(i)})(y_j^{(i)} - y^{(i)})^T \tag{6}$$

$$S_b = \frac{1}{N} \sum_{i=1}^{c} N_i (y^{(i)} - \overline{y})(y^{(i)} - \overline{y})^T \tag{7}$$

Then the modified Fisher criterion can be written as:

$$J(W) = \frac{\left| W^T S_b W \right|}{\left| W^T S_w W \right|} \tag{8}$$

The matrix W_{fld} which maximizes $J(W)$ is the direction to project.

The adaptively weighted Fisherface can modify the center of each class, so it's useful in classification with NCC (Nearest Cluster Center) classifiers. It also concentrates the distribution of the samples within each class and gets better S_w, S_b and a better projection matrix, so it's also useful with SVM classifiers.

3. Support Vector Machines for face recognition

3.1 Basic theory of Support Vector Machines

Support Vector Machines (SVMs) has been recently proposed by Vapnik and his co-workers [5， 10]. It plays an active role in the field of machine learning, data mining and so on. SVMs belongs to the class of maximum margin classifiers [6，11]. It performs pattern recognition between two classes by finding a decision surface that has the maximum distance to the closest points in the training set which are called support vectors. An example of OSH (Optimal Separating Hyperplane) and support vectors is shown in Figure.2

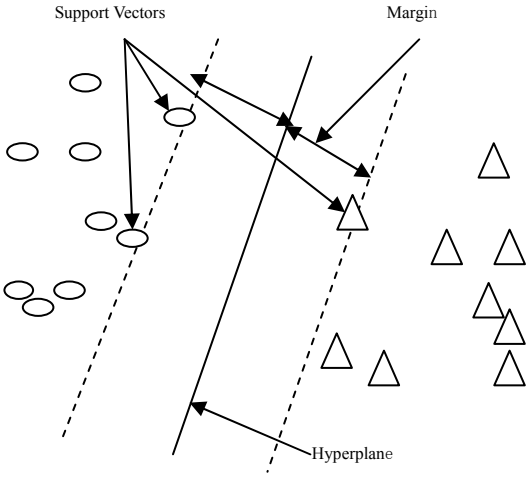


Figure. 2 The OSH with the largest margin, passing three support vectors

Supposing that we have a training set: $x_i, i = 1, \cdots, l$. Considering a simple case with two classes, we can define a vector y:

$$y_i = \begin{cases} 1 & if \ \ x_i \ in \ class \ 1 \\ -1 & if \ \ x_i \ in \ class \ 2 \end{cases} \tag{9}$$

The training set can be said optimally separated if it is separated without any error and the margin is the maximum [7]. The separating hyperplane must satisfy the following constrain:

$$y_i \left[(w \cdot x_i) + b \right] \geq 1, i = 1, 2, \cdots, l \tag{10}$$

The distance between a sample point x_i and the hyperplane is:

$$d(w, b; x_i) = \frac{\left| w \cdot x_i \right|}{\left\| w \right\|} \tag{11}$$

According to this, the distance between $w \cdot x + b = 1$ and $w \cdot x + b = -1$ is $\frac{2}{\left\| w \right\|}$, namely the margin equals to $\frac{2}{\left\| w \right\|}$.

In order to maximize the margin, we can minimize

$$\Phi(w) = \frac{1}{2} \left\| w \right\|^2 \tag{12}$$

We can solve the optimization problem of (12) under the constraint of (10) by calculating the following Lagrange function:

$$L(w,b,\alpha)=\frac{1}{2}\|w\|^2-\sum_{i=1}^{l}\alpha_i\left\{y_i\left[(w\cdot x_i)+b\right]-1\right\} \quad (13)$$

Where α_i denotes Lagrange multipliers. The minimization of this function can change into its dual mode:

$$\max_{\alpha} W(\alpha)=\max_{\alpha}\left\{\min_{w,b} L(w,b,\alpha)\right\} \quad (14)$$

The solution of (14) is given by:

$$\bar{\alpha}=\arg\min_{\alpha}\sum_{i=1}^{l}\alpha_i-\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j x_i\cdot x_j \quad (15)$$

It can be solved with two constrains:

$$\alpha_i\geq 0, i=1,\cdots,l \quad (16)$$

$$\sum_{i=1}^{l}\alpha_i y_i=0 \quad (17)$$

Since the Lagrange multipliers α_i have been solved out, the parameters in OSH can be determined also:

$$\bar{w}=\sum_{i=1}^{l}\bar{\alpha}_i y_i x_i \quad (18)$$

$$\bar{b}=-\frac{1}{2}\bar{w}\cdot\left[x_p+x_n\right] \quad (19)$$

Where x_p refers to the support vectors in positive samples, x_n refers to the support vectors in negative samples.

For a test sample x_t, we can classify it by:

$$f(x_t)=sign(w\cdot x_t+b) \quad (20)$$

The discussion above is the solution to separable case. In a non-separable case, we should just introduce slack variables ξ_i to allow a few errors in classification. The solution can be gotten in a similar way of separable case.

If the problem is nonlinear, the separating hyperplane can't be found no matter how hard we try. In order to solve the problem, the researchers thought that whether the training set was linear separable in other space? The answer can be found in [6], that is projecting the vectors of the training set into a higher space using a kernel function, so as to make them linear separable. There are 4 basic kernel functions [3]:

1. *linear*: $K(x_i,x_j)=x_i^T x_j$
2. *polymomial*: $K(x_i,x_j)=(\gamma x_i^T x_j+r)^d, \gamma>0$
3. *radial basis function*(RBF):

$$K(x_i,x_j)=\exp(-\gamma\left\|x_i-x_j\right\|^2), \gamma>0$$

4. *sigmoid*: $K(x_i,x_j)=\tanh(\gamma x_i^T x_j+r)$

Here, γ, r, d are kernel parameters.

3.2 Training set optimization

We can see that the separating hyperplane is determined by support vectors. That means support vectors play a significant role in the course of finding the OSH, while the other points in the training set contribute little. But the whole training set which includes a high percent of non-support vectors is involved during calculating the SVM model. Training SVM is an optimization course. It includes not only the optimization of support vectors, but also the optimization of non-support vector points. So the time spent on these inessential points is regarded as unnecessary. It's really a waste of time when the training set is large. These points can be ignored without losing the classification ability of SVM [8，9]. The CDR (Center Distance Ratio) was proposed to deal with the unwanted points in the training set. We define d_1 as the distance between a certain point and the center of the class which it belongs to, d_2 as the distance between the point and the center of the opposite class. Both d_1 and d_2 are calculated in the high-dimension space projected by kernel function. The CDR is:

$$R=\frac{d_1}{d_2} \quad (22)$$

d_1 and d_2 can be determined by the following:

$$d_1(x,c_{own})=\sqrt{K(x,x)-\frac{2}{n_1}\sum_{i=1}^{n_1}K(x,x_i)+\frac{1}{n_1^2}\sum_{i=1}^{n_1}\sum_{j=1}^{n_1}K(x_i,x_j)} \quad (23)$$

$$d_2(x,c_{opp})=\sqrt{K(x,x)-\frac{2}{n_2}\sum_{i=1}^{n_2}K(x,x_i')+\frac{1}{n_2^2}\sum_{i=1}^{n_2}\sum_{j=1}^{n_2}K(x_i',x_j')} \quad (24)$$

Where c_{own} is the center of the class that x belongs to, c_{opp} is the center of the opposite class, and n_1 and n_2 refer to the number of points in the two classes. We can understand it easily from Figure.3.

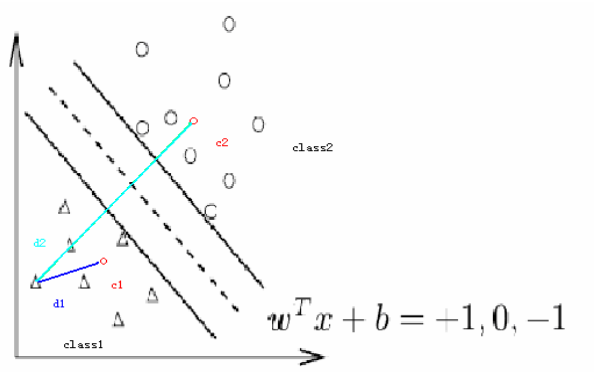


Figure.3 The distance between a point and the centers of the two classes

It can be seen that the points which have smaller CDR lie farther away from the separating hyperplane than the ones with a larger CDR. The points which lie near the separating hyperplane are more possible the be the support vectors. So we can pick out the points with larger CDR from each class to make a better and condensed training set. The size of the training set can even be cut to 20% if the data are good enough.

3.3. Kernel parameters determination

The classification ability of SVM is sensitive to kernel parameters. Here, we use a popular and efficient way to get the best kernel parameters: Cross-Validation（an example of C-V is shown in Table 1）. The main idea of C-V is LOO (Leave one out). Firstly, divide the training set into *k* folds of the same size randomly. Choose one fold to be the test data, the other *k-1* folds to be the training data, and get the accuracy of classification of the SVM which is trained by the *k-1* folds. Then choose another fold to be the test data and do the same as what mentioned above.

There also exists a big problem in the training course of SVM: Overfitting. If we use all the training data at the same time to find the kernel parameter which can bring the lowest training error rate, we may encounter the overfitting problem. So even the SVM has a high training accuracy, it has a low generalization ability (see Figure.4)[12]. And Cross-Validation is a good way to avoid this.

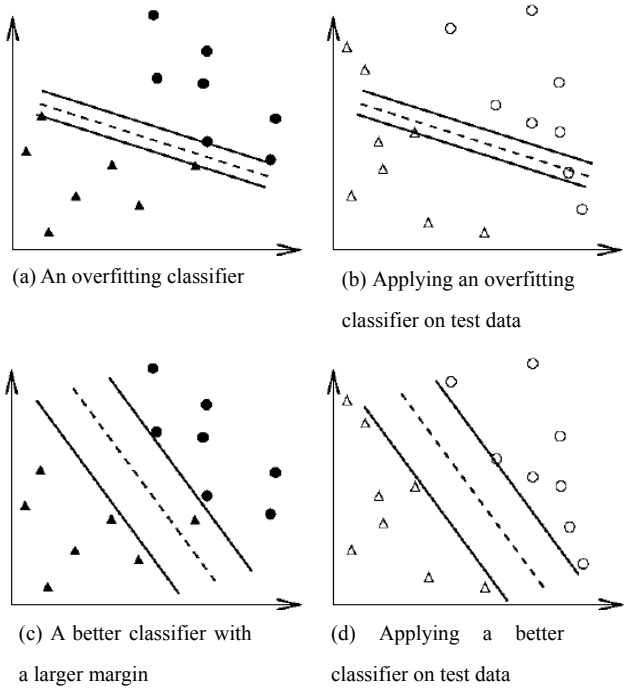
Table 1. A complete course of 3-folds Cross-Validation

	Test samples	Training samples	Predict accuracy
Round 1	1	2,3	Acc1
Round 2	2	1,3	Acc2
Round 3	3	1,2	Acc3

The accuracy of k-folds C-V is defined as below:

$$Accuracy_{c-v} = \frac{1}{k}\sum_{i=1}^{k} Acc_i \qquad (25)$$

Here, Acc_i means the classification accuracy of the *i*st round.



(a) An overfitting classifier

(b) Applying an overfitting classifier on test data

(c) A better classifier with a larger margin

(d) Applying a better classifier on test data

Figure.4 The overfitting problem. (a) and (c) show different classifiers trained by the same training data, the classifier in (a) has a lower training error rate, but with a smaller margin, while the classifier in (c) has a higher training error rate, but with a larger margin. By comparing the classification results in (b) and (d), we can find that an overfitting classifier usually has a low generalization ability, even with a high training accuracy rate.

We use Grid Search to search for the optimized parameters in the space consisted of the possible value of them. Parameters with best C-V accuracy are taken as the kernel parameters of the SVM.

3.4 Multi-class Recognition

SVM can only handle two-class cases. If there are more than two classes in our task, we should use some strategies to extend it from two-class to multi-class. Three ways are used the most frequently for this purpose. One is the one-against-all (O-A-A) strategy to classify between one class and the left classes. Another is the one-against-one (O-A-O) strategy to classify between each class. The third one is Binary tree structure SVMs [7].

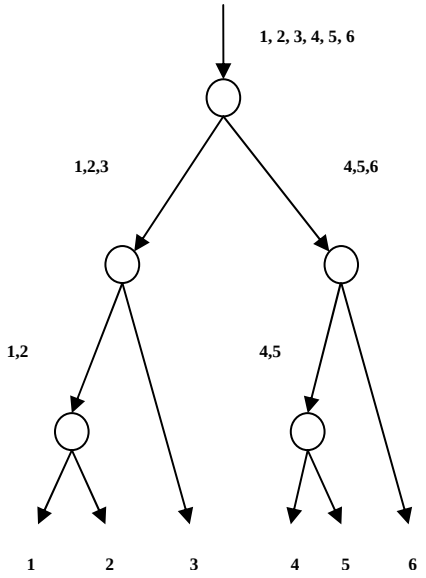


Figure. 5 The structure of a 6-class binary tree SVMs

All of the three strategies have their own advantages and disadvantages. O-A-A is the simplest of them, it treats one class as positive samples and the left classes as negative. We need n SVMs for a n-class classification by O-A-A. Because the size of positive and negative samples varies large, we may meet an unbalanced-data problem during the use of it. The one-against-one structure has a high accuracy because of training SVMs between each class. We'll have $\frac{n(n-1)}{2}$ SVMs to solve an n-class case. The classification result of the test data is voted by the $\frac{n(n-1)}{2}$ SVMs. When n is large, the number of SVMs may become horrible. Here we choose the binary tree structure SVMs to implement n-class classification with only $n-1$ SVMs. An example of 6-class binary tree is shown in Figure.5

4. Experimental results

Our experiment is on the Cambridge Olivetti Research Lab (ORL) face database. The database contains 400 images of 40 individuals. We choose 100 images each time from it as our training data, 20 individuals, 5 images each. The left images from the 20 individuals are used as test images. The image database includes images that varies in poses (smiling or peaceful, eyes open or closed), heading directions, and details (wear glasses or no glasses). The task of face recognition from this database is really a difficult mission. Especially the number of people to be recognized is up to 20, and the number of training image of each people is only 5. The experiment was carried on PC with Pentium Centrino 1.5 MHz, 768 M ram and Windows operation system. We use libsvm toolbox [12] for all the SVMs. We do 4 experiments on images of different people chosen from the database. The results are shown in the following table:

Table 2. Four experiments classification accuracy results (feature dimension:20, β :0.1)

Experiment	1	2	3	4
Accuracy	83%	86%	86%	89%


Figure. 6 4 of 20 people's images in the training database, 5 for each

From the images shown in Figure. 6 we can see, images of the same person in the database may be quite different from each other (some images may have a difference up to 90° in the direction of heading).

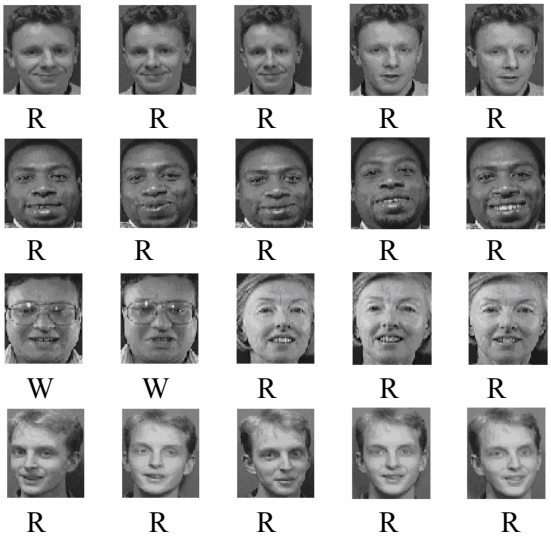


Figure. 7 the recognition result of the 4 people in experiment 4. The capital letter "R" means that the recognition result of the test image is correct, while "W" refers to wrong.

We get an average classification accuracy of 86% on the total 400 test images (4 experiments). 4 of 20 people's recognition result of experiment 4 is shown in Figure.7. From Figure.7 we can also get a satisfactory result with only 2 errors in the 4 people's test images. It can be proved that our approach is robust to poses and face expressions.

We perform experiment 4 with different feature vector dimensions, and the curve of recognition accuracy is shown in Figure.8

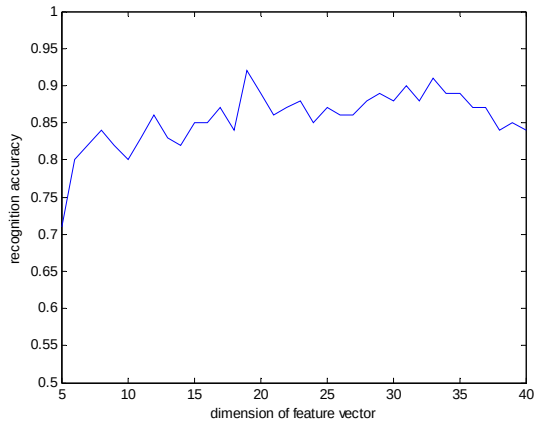


Figure. 8 The curve of recognition accuracy of experiment 4 with different dimensions of feature vector.

To illustrate the benefit of adaptively weighted Fisherface, we compare the recognition result with that of conventional Fisherface on experiment 4 using different dimensions of feature vector. It proves that adaptively weighted Fisherface is better than conventional Fisherface from Figure. 9

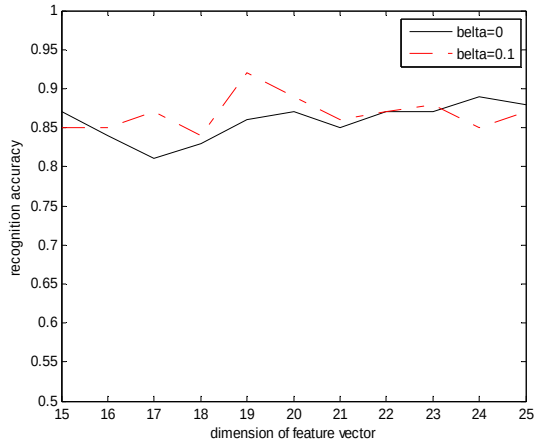


Figure. 9 A comparision of recognition results on experiment 4 between adaptively weighted Fisherface (β =0.1) and the conventional Fisherface (β =0)

5. Conclusions

The average classification accuracy of 86% (with 20 dimension feature vector) is an acceptable result, with a low computational cost on feature extraction and SVM modeling. Many SVM tools use the one-against-one structure to deal with multi-class classification. There are 20 classes in our experiment, so we should train 190 SVMs using this structure. It will induce a high consumed cost in computation. In our approach, only 19 SVMs are needed to be trained, just 10% of the former approach. And the time spending is also quite low. The recognition process is real-time. From Figure.6 we can see that the curve of recognition accuracy is relatively stable, it means that our approach can also get a good result even with a few features. It's really a delighted property in the situation when feature extraction is very hard. The recognition curve reaches the maximum of 92% at the dimension of 19. The whole flow of our method on face recognition is proved valid and efficient. We can expect a higher classification accuracy by modeling the face images with a more complex method, such as HMM [1] and better SVM training, such as a more precise algorithm in searching for kernel parameters. These are the aims of our research in the future.

Acknowledgement

We would like to thank for the support from Open Fund of State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing of China, and The National High Technology Research and Development Program of China（No.2007AA12Z155）

References

[1]Nuria M. Oliver, Barbara Rosario, and Alex P. Pentland: "A Bayesian Computer Vision System for Modeling Human Interactions" Vol. 22, No. 8, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Aug. 2000.

[2]Peter N. Belhumeur, Joao P. Hespanha, David J. Kriegman: "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection", *ECCV*, 1996.

[3]Cortes, C. and V. Vapnik: "Support-vector network" *Machine Learning* 20,273–297,1995.

[4]Yin Hong-tao, Fu Ping, Meng Sheng-wei: "Face Recognition based on Adaptively Weighted Fisherface" *Journal of Optoelectronics & Laser*, Vol.17, No.11, pp.1406~1408, Nov. 2006

[5] V. N. Vapnik: "Statistical learning theory". John Wiley & Sons, New York, 1998.

[6]Bernd Heisele, Purdy Ho, Tomaso Poggio: "Face Recognition with Support Vector Machines: Global versus Component based Approach" In*: Proc. Of ICCV*, 2001.

[7]G. Guo, Stan Z. Li, and Kapluk Chan: "Face Recognition by Support Vector Machines" *Digital Object Identifier* 10.1109/AFGR, 2000.
[8] L Zhang, WD Zhou, LC. Jiao. "Pre-extracting Support Vectors for Support Vector Machine". In: .*IEEE Proceedings of ICSP2000*, pp.1432 – 1435, 2000.
[9]Osuna E, Freund R, Girrosi F: "Improved Training Agorithm for Support Vector Machines", In: *IEEE Pro. NNSP*, New Jersey, 1997.
[10]K. Jonsson, J. Kittler, Y. P. Li, and J. Matas. "Support Vector Machines for Face Authentication". In T. Pridmore and D. Elliman,editors, *BMVC'99*, pp. 543–553, 1999.
[11]C. J. C. Burges. "A Tutorial on support vector machines forpattern recognition". *Data Mining and Knowledge Discovery*, Vol. 2,No.2, pp.121–167, 1998.
[12]http://www.csie.ntu.edu.tw/~cjlin/