Contents lists available at SciVerse ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

INFORMATION SCIENCES

Using the idea of the sparse representation to perform coarse-to-fine face recognition

Yong Xu[a,b,*], Qi Zhu[a], Zizhu Fan[a,c], David Zhang[d], Jianxun Mi[a], Zhihui Lai[a]

[a] *Bio-Computing Research Center, Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, China*
[b] *Key Laboratory of Network Oriented Intelligent Computation, Shenzhen, China*
[c] *School of Basic Science, East China Jiaotong University, Nanchang, Jiangxi, China*
[d] *Biometrics Research Centre, Department of Computing, Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong*

ARTICLE INFO

Article history:
Received 4 May 2010
Received in revised form 17 February 2013
Accepted 27 February 2013
Available online 7 March 2013

Keywords:
Biometrics
Security access
Face recognition
Information fusion
Decision making

ABSTRACT

In this paper, we propose a coarse-to-fine face recognition method. This method consists of two stages and works in a similar way as the well-known sparse representation method. The first stage determines a linear combination of all the training samples that is approximately equal to the test sample. This stage exploits the determined linear combination to coarsely determine candidate class labels of the test sample. The second stage again determines a weighted sum of all the training samples from the candidate classes that is approximately equal to the test sample and uses the weighted sum to perform classification. The rationale of the proposed method is as follows: the first stage identifies the classes that are "far" from the test sample and removes them from the set of the training samples. Then the method will assign the test sample into one of the remaining classes and the classification problem becomes a simpler one with fewer classes. The proposed method not only has a high accuracy but also can be clearly interpreted.

1. Introduction

Biometrics is one of the most important branches of pattern recognition [8,12,13,31,36,38]. Face recognition is one of the most attractive biometric techniques. Nevertheless, face recognition is still a challenging task [2,16–18,32,39,44]. This is mainly owing to varying lighting, facial expression, pose and environment [37]. We know that the transform method is a kind of computationally efficient method for face recognition. Typical transform methods include linear [3,21,22,25,27] and nonlinear transform methods [5–7,9,23,24,41]. A common procedure of the transform methods such as principal component analysis (PCA) [10,11,43] and linear discriminant analysis (LDA) [4,15] is to first exploit all the training samples to produce the transform axes, and then use the produced transform axes to transform all the samples into a lower-dimensional space. We refer to those transform methods that use all the training samples to implement the training procedure as global transform learning methods.

A number of learning methods using local transform have also been proposed in recent years. When performing the transformation or learning, these methods exploit only the information of a subset of the training samples. Examples of these methods include the methods proposed in [26,28,37,47]. In Harandi et al. [26], attempted to obtain the optimal local bases that are beneficial to find the most discriminant features for different parts of the face space. Indeed, they obtained optimal local bases by learning the local information of training samples. Harandi et al. [26] showed that their proposed method outperformed

[*] Corresponding author at: Bio-Computing Research Center, Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, China.
E-mail address: laterfall2@yahoo.com.cn (Y. Xu).

0020-0255/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2013.02.051

most of the previous approaches which solve the recognition problem by using a single basis for all individuals. In Sugiyama [28], proposed a transform method to integrate the ideas of LDA and locality preserving projection (LPP). Sugiyama showed that the proposed method was very suitable for the problem in which samples in a class were multimodal, i.e., samples from the same class form several separate clusters [28]. In Vural et al. [37], exploited the local dependencies of samples for classification. In Liu et al. [47], proposed local-PCA-based feature extraction method. Literatures [1,14,33,46] also proposed several methods to use the relationship between samples in a local region. In this paper, we refer to all of these methods as local methods. Hereafter "local" means that the method exploits only a subset of the training samples rather than all the training samples to classify each test sample.

Recently, researchers proposed a special kind of face recognition method, i.e., sparse representation method (SRM) [19,20]. This method uses a linear combination of all the training samples to "sparsely" represent and classify the face image. Here "sparsely" means that some coefficients of the linear combination are equal or close to zero. In SRM, the difference between the test sample and the weighted sum of the training samples of a class is referred to as the representation residual of this class. The coefficient of a training sample in the linear combination acts as the weight of this training sample. The method assigns the test sample into the class that produces the minimum representation residual [19,20].

In this paper, motivated by local transform methods and the sparse representation methods, we propose a new face recognition method. This method exploits the training samples of a small number of classes that are "close" to the test sample to classify it. Because this method uses only a subset of all the training samples to represent the test sample, we can also view the method as a sparse representation method. The proposed method includes two stages. The first stage identifies the classes that are "close" to the test sample and removes the training samples of the other classes from the set of training samples. Then the second stage represents the test sample as a linear combination of all the training samples from the remaining classes. The second stage also determines the contributions of the training samples of each individual class in representing the test sample, and exploits the contribution to classify the test sample. In this work, we devise two algorithms for the first stage. The first algorithm assumes that the test sample can be approximately expressed as a linear combination of all the training samples and uses the determined linear combination to identify the classes that are close to the test sample. The second algorithm for the first stage directly exploits the Euclidean distance to identify the classes that are close to the test sample.

The proposed method has the following rationale: as the classes in the second stage are only a subset of all the original classes, the classification problem will become simpler. In other words, the original classification problem needs to assign the test sample into one of all the original classes, whereas the classification problem in the second stage of the proposed method just needs to assign the test sample into one of few classes. Usually, to assign the test sample into one of few classes is simpler and will obtain higher accuracy. In this paper, the class label is denoted by an integer. The class with class label "i" is also called the ith class. Moreover, as shown in Section 3.4, the rationale of the proposed method can also be clearly interpreted. The experimental results illustrate the good performance of our method.

The remainder of this paper is organized as follows: in Section 2, we describe our method. Section 3 shows the rationale and interpretation of our method. Section 4 presents the experimental results. Finally, Section 5 offers our conclusion.

2. The coarse-to-fine face recognition (CFFR) method

This section describes the proposed coarse-to-fine face recognition (CFFR) method in detail. The code of CFFR can be downloaded at http://www.yongxu.org/lunwen.html. Suppose that there are L classes and n training samples, $x_1, \ldots, x_n$. Hereafter each sample is assumed to be a one-dimensional column vector.

2.1. The first stage of CFFR

The first stage of CFFR determines a linear combination of all the training samples that is approximately equal to the test sample. Based on the combination, we can coarsely determine the candidate class labels of the test sample. This stage assumes that the following equation is approximately satisfied:

$$y \approx a_1 x_1 + \cdots + a_n x_n, \tag{1}$$

where y is the test sample and a_i ($i = 1, 2, \ldots, n$) are the coefficients of x_i. Eq. (1) means that the test sample can be approximately represented by a linear combination of all the training samples. By replacing "$\approx$" by "=", we transform Eq. (1) into the following equation:

$$y = XA, \tag{2}$$

where $X = [x_1 \cdots x_n]$, $A = [a_1 \cdots a_n]^T$. We solve A using $\widetilde{A} = (X^T X + \mu I)^{-1} X^T y$, where μ is a small positive constant and I is the identity matrix.

Eq. (1) also shows that different training samples have different contributions in representing the test sample. For example, the contribution of the ith training sample is $\tilde{a}_i x_i$ where $\tilde{a}_i$ is the ith component of $\widetilde{A}$. The test sample can be approximately represented by the sum of the contributions of the training samples from all the classes. We also measure the contribution in expressing the test sample of the kth class using

$$e_k = \left\| y - \sum_{i=p}^{q} \tilde{a}_i x_i \right\|^2. \tag{3}$$

e_k is referred to as the residual of the kth class with respect to y. A small e_k means that the kth class ($k \in C$) has great contribution in representing the test sample. $x_p, x_{p+1}, \ldots, x_q$ denote all the training samples from the kth class, and $\tilde{a}_p, \tilde{a}_{p+1}, \ldots, \tilde{a}_q$ stand for the pth, $p + 1$th and qth components of $\widetilde{A}$, respectively.

Let $e_{q_1}, e_{q_2}, \ldots, e_{q_L}$ stand for the residuals of the classes with class labels $c_{q_1}, c_{q_2}, \ldots, c_{q_L}$, respectively. We assume $e_{q_1} \leqslant e_{q_2} \cdots e_{q_L}$. CFFR takes the s class labels with the first s smallest residuals, i.e. $c_{q_1}, c_{q_2}, \ldots, c_{q_s}$, as the candidate classes for the test sample. The training samples from the s candidate classes are reserved, and all the training samples from the other classes are eliminated from the training set.

2.2. The second stage of CFFR

The second stage of CFFR first represents the test sample as a linear combination of the training samples of the first s candidate classes, and then exploits this linear combination to classify the test sample. This stage assumes that

$$y \approx d_1 t_1 + \cdots + d_M t_M, \tag{4}$$

where M is the number of all the training samples from the first s candidate classes, t_i ($i = 1, 2, \ldots, M$) denote the training samples of all these classes, and d_i ($i = 1, 2, \ldots, M$) is the coefficient of t_i. We rewrite Eq. (4) as

$$y = GD, \tag{5}$$

where $D = [d_1 \cdots d_M]^T$, $G = [t_1 \cdots t_M]$. We obtain the solution of D using

$$\widetilde{D} = (G^T G + \gamma I)^{-1} G^T y, \tag{6}$$

where γ is a small positive constant and I also denotes the identity matrix. We refer to $G\widetilde{D}$ as the representation result of the test sample, obtained using CFFR. The representation result can also be transformed into a two-dimensional image with the same size as the original sample image.

The classification rule of the second stage of CFFR is as follows: first, the sum of the contribution in representing the test sample of the rth class is calculated using

$$f_r = \tilde{d}_g t_g + \cdots + \tilde{d}_h t_h, \tag{7}$$

where $t_g \cdots t_h$ stand for all the training samples of the rth class ($r \in C = \{c_{q_1}, c_{q_2}, \ldots, c_{q_s}\}$). $\tilde{d}_g \cdots \tilde{d}_h$ denote the coefficients of $t_g \cdots t_h$, respectively. $\tilde{d}_g \cdots \tilde{d}_h$ are also the gth, ..., the hth entries of $\widetilde{D}$, respectively. f_r can also be converted into a two-dimensional matrix with the same size as the original sample image. The obtained matrix is referred to as the two-dimensional image of the contributions of the rth class. We calculate the ultimate residual of the rth class with respect to test sample y using

$$de_r = ||y - f_r||^2, \quad r \in C. \tag{8}$$

The second stage of CFFR classifies y into the class that produces the smallest ultimate residual.

2.3. Summary of CFFR

The main steps of the algorithm of CFFR can be summarized as follows:

Step 1. Solve Eq. (2).
Step 2. Determine the s candidate class labels $c_{q_1}, c_{q_2}, \ldots, c_{q_s}$ of the test sample based on Eq. (3).
Step 3. Obtain a linear combination of all the training samples from the c_{q_1}th, c_{q_2}th, ..., and c_{q_s}th classes that is approximately equal to the test sample. That is, to use (6) to solve Eq. (5).
Step 4. Calculate ultimate residual de_r using Eq. (8). A small de_r means that the rth class ($r \in C$) has a great capability in representing the test sample.
Step 5. Identify the class that has the minimum ultimate residual and classify the test sample into this class.

Fig. 1 describes the flowchart of CFFR. It is clear that the first stage of CFFR consists of Steps 1 and 2 and the second stage of CFFR consists of Steps 3, 4 and 5.

3. Interpretation and underlying rationale of CFFR

This section aims at showing the underlying rationale and giving an interpretation of CFFR. This section also shows an alternative scheme of CFFR.

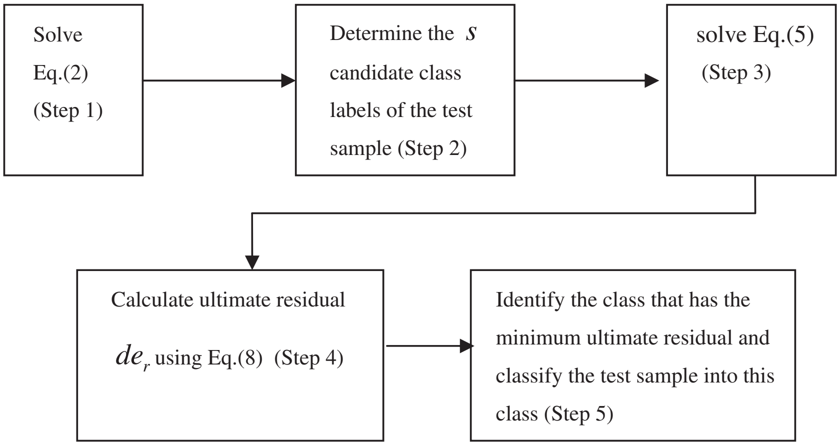
Solve Eq.(2) (Step 1)
Determine the s candidate class labels of the test sample (Step 2)
solve Eq.(5) (Step 3)
Calculate ultimate residual de_r using Eq.(8) (Step 4)
Identify the class that has the minimum ultimate residual and classify the test sample into this class (Step 5)

Fig. 1. Flowchart of CFFR. The first stage of CFFR consists of Steps 1 and 2, and the second stage of CFFR consists of Steps 3, 4 and 5.

3.1. Justification and rationale of CFFR

CFFR has the following justification and rationale: though the training samples of all the classes are available for devising the classification algorithm, all the training samples do not have the same effect on the classification decision of the test sample. It is reasonable to assume that the class close to the test sample has great effect, and if one class is far enough from the test sample it will have little effect and even have side-effect on the classification decision of the test sample. As a result, when devising the classification algorithm, we can first exclude the classes that are very far from the test sample and depend on only the remaining classes to make the classification decision. If we do so, we can eliminate the side-effect on the classification decision of the class that is far from the test sample. As a result, we may obtain higher classification accuracy.

The second stage of CFFR needs to assign the test sample to one of the s candidate class labels, thus it is indeed faced with a problem simpler than the original classification problem, which needs to assign one of the original $L(L > s)$candidate class labels to the test sample. Actually, when the first stage of CFFR determines the s candidate class labels for the test sample, it has converted the L-class problem into an s-class problem. Usually, for a classification problem, the more the classes are, the lower the maximum possible classification accuracy is.

3.2. Relationship with the sparse representation

In this subsection we will show that CFFR can be also viewed as a supervised sparse representation method and analyze its computational complexity. We first simply describe the sparse representation methods proposed in [19,20]. The method proposed in [19,20] represented the test sample as a sparse linear combination of all the training samples and obtained promising performance for face recognition. In this paper, the "sparseness" is defined as follows: when the method determines a linear combination of all the training samples that is approximately equal to the test sample, the coefficients of some training samples are equal or close to zero. We also refer to these coefficients as sparse coefficients.

CFFR can be somewhat viewed as a supervised sparse representation method. This is because when we rewrite the linear combination in the second stage of CFFR as a linear combination of all the original training samples, the coefficients of the training samples that have been removed from the original set of the training samples must be zero. Here, "supervised" means that in CFFR it is known how sparse the coefficients are (i.e., there are how many zero coefficients) and which coefficients are zero. However, for the sparse representation method proposed in [20], it is not clearly known which coefficients of the linear combination are equal or close to zero. In addition, as the method proposed in [19,20] used an iterative scheme to obtain the numerical solution, it is not guaranteed that the "sparse" coefficients are true zero. However, it is sure that the sparse coefficients of our method are true zero. Moreover, the supervised sparse representation method proposed in this paper provides a very simple and feasible way to lead to sparse representation, i.e., to make the classes that are "far" from the test sample have zero coefficients. Our work also visually illustrates that the sparse representation is really beneficial to classification. Actually, as shown in the experimental section, compared with the proposed method, the global version of CFFR, i.e., the all-training-samples-based non-sparse representation method, almost always obtains a lower accuracy.

CFFR is also as a computationally efficient sparse representation method. The first and second stages of CFFR need to solve only two linear systems, whereas the method proposed in [19,20] has a much higher computational cost. Thus, it is clear that our method is computationally much more efficient than the method proposed in [19,20]. We analyze the computational cost of CFFR as follows. In the first stage of CFFR, if every sample is a $K \times 1$ column vector, then $B = X^TX$ has a computational cost of $o(Kn^2)$. It is clear that the computational cost of $W = (B + \mu I)^{-1}$ is $o(n^3)$. Moreover, WX^Ty has a computational cost of $o(Kn^2) + o(Kn)$. As a result, the first stage of CFFR has computational cost of $o(n^3)$+$o(Kn^2) + o(Kn)$in total. Since $Kn \ll Kn^2$, we might say that the computational cost of the first stage of CFFR is $o(n^3) + o(Kn^2)$. In a similar way, we know that the second stage of CFFR has a computational cost of $o(M^3) + o(KM^2)$. As a result, CFFR needs a computational cost of $o(n^3) + o(Kn^2)+o(M^3) + o(KM^2)$.

3.3. Alternative scheme of CFFR

In this section we present an alternative scheme of CFFR (ASCFFR). ASCFFR is also composed of two stages and its second stage is identical to that of CFFR. The first stage of ASCFFR is different from that of CFFR as follows: it uses the Euclidean distance to determine the s classes that are closest to the test sample and takes the class labels of these s classes as s candidate class labels of the test sample. The first stage of ASCFFR first calculates the Euclidean distance between the test sample and each training sample. Let $x_1 \cdots x_n$ still denote the n training samples. If $x_1^i \cdots x_{n_i}^i$ are the n_i training samples of the ith class, then the sum of the Euclidean distances between the test sample and these training samples is referred to as the distance between the test sample and the ith class. The class labels of the s classes that have the minimum s distances are denoted by the set $C' = \{c_{q_1}, c_{q_2}, \ldots, c_{q_s}\}$. ASCFFR takes $c_{q_1}, c_{q_2}, \ldots, c_{q_s}$ as s candidate class labels of the test sample.

The main steps of the algorithm of ASCFFR can be summarized as follows:

Step 1. Calculate the Euclidean distance between the test sample and each class.
Step 2. Determine the s candidate class labels of the test sample in terms of the distances obtained in Step 1.
Step 3. Use a linear combination of all the training samples from the s classes determined by Step 2 to approximate test sample y. That is, to use (5) to solve Eq. (5).
Step 4. Use Eq. (8) to calculate ultimate residual de_r, $r \in C'$.
Step 5. Identify the class that has the minimum ultimate residual and classify the test sample into this class.

3.4. Interpretation of the method

In this subsection, we will give the interpretation of ASCFFR (CFFR can be also interpreted in a similar way). The first stage of ASCFFR can be viewed as a stage that exploits the distances between the test sample and different classes to coarsely determine the posterior probability of the test sample, whereas the second stage of ASCFFR more accurately determines the posterior probability and ultimately classifies the test sample. If the distance between the test sample and the ith class is $dist_i$, then the posterior probability $p(c_i|y)$ can be defined as $p(c_i|y) \propto 1 - \frac{dist_i}{\sum_{j=1}^{L} dist_j}$, $i = 1, 2, \ldots, L$. As shown in Fig. 2, $p(c_i|y)$ might vary severely with i. This figure shows that there are six classes and the second and sixth classes have the first two largest posterior probabilities. We assume that the test sample is truely from the second class. In real-world applications, it is possible that the test sample is erroneously classified into the sixth class. For example, if among all the training samples, one sample from the sixth class is closest to the test sample and the nearest neighbor classifier is used for classification, then the test sample will be erroneously classified.

The first stage of ASCFFR also modifies the posterior probability as follows: if a class is far enough from the test sample and has a small posterior probability (shown by the red bar), the first stage of ASCFFR will not take into account its effect on the classification of the test sample, taking only the class labels of the other classes as the candidate class labels of the test sample. This is equivalent to the action to assign zero to $p(c_j|y)$, where c_j denotes a class that is "far" enough from the test sample. The first stage does not simply regard that the test sample is from the class that has the largest posterior probability.

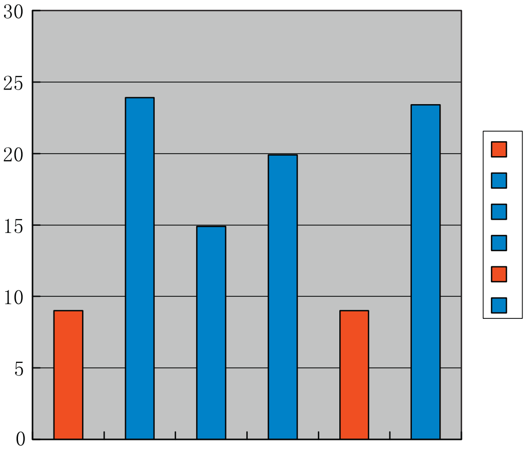

Fig. 2. A possible distribution of the original posterior probability $p(c_i|y)$, coarsely determined by the first stage of ASCFFR. The horizontal axis shows there are six classes and the vertical axis shows the posterior probability (%) of each class.

On the other hand, the first stage assumes that it is possible for the test sample to be from each of the classes whose posterior probabilities are not zero (or are not set to zero). Thus, we say that the first stage of ASCFFR performs coarse classification.

As $\sum_{i=1}^{L} p(c_i|y)$ should equal to 1 and the first stage sets the posterior probabilities of some classes to zero, the second stage of ASCFFR will increase the posterior probabilities of the other classes to make $\sum_{i=1}^{L} p(c_i|y)$ still equal to 1. Actually, the second stage of ASCFFR assigns new values to all $p(c_i|y)$, $c_i \in C$ as follows: it first assumes that posterior probability $p(c_i|y)(c_i \in C)$ is directly related to de_i, i.e., $p(c_i|y) \propto 1 - \frac{de_i}{\sum_i de_i}$, $c_i \in C$. It is clear that the smaller de_i, the greater $p(c_i|y)$. Finally, the ASCFFR classifies the test sample into the class with the greatest posterior probability. Fig. 3 visually depicts a possible distribution of the posterior probability determined by the second stage of ASCFFR and shows that ASCFFR can correctly classify the test sample into the second class. Fig. 3 indeed shows a modification of the posterior probability shown in Fig. 2. Fig. 4 summarizes the interpretation of ASCFFR.

4. Experimental results

We conducted a number of experiments on the ORL [35], Feret [29,30] and AR [34] face databases. From the AR face database, we used 3120 gray face images from 120 subjects, each providing 26 images. These images were taken in two sessions [34]. Fig. 5 shows some face images from the AR database. From the Feret face database, we used only a subset made up of 1400 images from 200 individuals and each one contains seven images [40]. This subset was composed of images whose names are marked with two-character strings: ‘ba’, ‘bj’, ‘bk’, ‘be’, ‘bf’, ‘bd’, and ‘bg’. For each of the ORL and Feret databases, if t samples drawn from a subject are used for training samples (suppose that a subject contains m samples), there are $C_m^t = \frac{m(m-1)\ldots(m-t+1)}{t(t-1)\ldots1}$ possible cases. In other words, there are C_m^t ways to select t samples as training samples from all the m samples. We used the same C_m^t ways to select training samples from the samples of every subject and took the remaining samples of every subject as test samples. As a result, we obtained C_m^t training sets and C_m^t testing sets. For the Feret database, we set $t = 4$. As a result, there were 35 training and 35 testing sets. We used all the sets to test the methods. For the ORL database, we performed two experiments. In the first experiment we set $t = 5$, so there were 252 training sets and the same number of testing sets. In the second experiment we set $t = 6$ and there were 210 training sets and the same number of testing sets. For the AR face database, we used only one training set and testing set to conduct the experiment. Specifically, we used the first eight samples per subject in the AR face database as training samples and took the others as test samples. We resized each face image of the AR database to a 40 by 50 image by using the down-sampling algorithm presented in [45]. The face images of the ORL and Feret databases were also resized using the same algorithm. Before all the methods were carried out, each face image was converted into a one-dimensional column vector that has unit L_2 norm in advance. When solving Eqs. (2) and (5), we set both of μ and γ to 0.01. We used software ‘Matlab’ to implement all the methods. All the experiments were run on an Intel Xeon X3430 PC.

In order to make readers easily understand Figs. 6 and 7 shown later, we interpret f_r in Eq. (7) as follows. f_r can be viewed as the reconstruction result of the test sample, generated from the rth class. As presented in Section 2.2, if among all f_r, f_s has the smallest residual with respect to test sample y, then CFFR classifies the test sample into the sth class. We can convert f_r into a matrix I_r with the same size as the face images used in the experiment. We refer to I_r as the reconstruction image of the test sample, generated from the rth class. In the face recognition problem, because I_r is indeed a linear combination of the training samples of the rth subject, it is clear that I_r looks like a face image of the rth subject. As a result, if the test sample is truly from the sth subject and this subject has the smallest residual among all the subjects, the method will correctly classify the test sample. Moreover, the reconstruction image generated from the sth subject will look like the original image of the same subject.

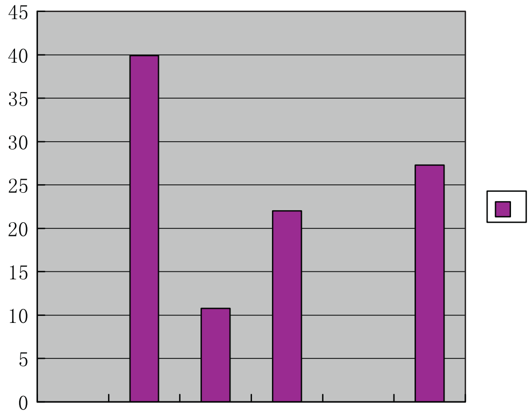

Fig. 3. A possible distribution of the posterior probability $p(c_i|y)$ determined by the second stage of ASCFFR. The horizontal axis shows there are six classes and the vertical axis shows the posterior probability (%). The posterior probability shown in this figure is a modification of the posterior probability shown in Fig. 2. In this figure, both the first and fifth classes have a posterior probability of zero.

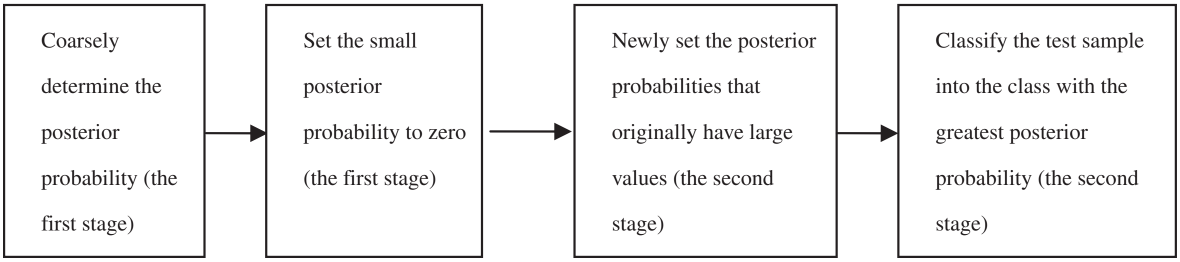
Coarsely determine the posterior probability (the first stage)
Set the small posterior probability to zero (the first stage)
Newly set the posterior probabilities that originally have large values (the second stage)
Classify the test sample into the class with the greatest posterior probability (the second stage)

Fig. 4. Flowchart of ASCFFR.

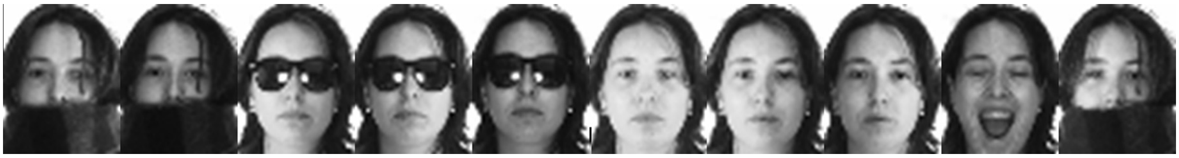

Fig. 5. Some face images of one subject from the AR database.

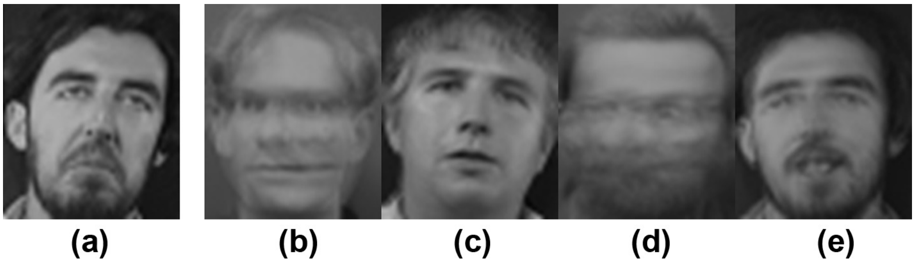

Fig. 6. An original test sample and the reconstruction images obtained using the global version of CFFR. (a) Shows the original test sample from the ORL database. (b–e) Show the reconstruction images of the four classes that have the first four smallest ultimate residuals, respectively. It is clear that since the reconstruction image with the smallest ultimate residual (shown in b) was not generated from the subject of the test sample, the global version of CFFR would not correctly classify the test sample.

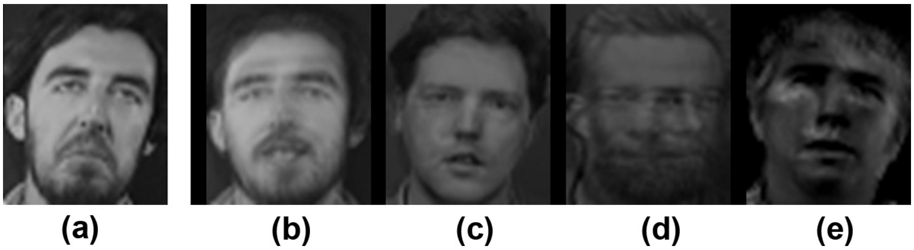

Fig. 7. The same test sample shown in Fig. 6 and the reconstruction images obtained using CFFR. In CFFR s was set to 10. (a) Shows the same original test sample shown in Fig. 6. (b–e) Show the reconstruction images of the four classes that have the first four smallest ultimate residuals, respectively. It is clear that CFFR correctly classified the test sample.

We use Figs. 6 and 7 to show one original test sample and the reconstruction images. Figs. 6 and 7 respectively show that the results of CFFR and the global version of CFFR, on one test sample from the ORL database. Hereafter the global version of CFFR means that CFFR is carried out in the following way: it first represents the test sample by a linear combination of all the training samples of all the classes and then it calculates e_k using Eq. (3), and classifies the test sample into the class with the minimum e_k. In other words, the global version of CFFR indeed implements only the first stage of CFFR and does not carry out the second stage of CFFR at all. Clearly, the global version of CFFR is identical to the global representation method presented in [42]. Moreover, from Section 3 we know that the global versions of ASCFFR and CFFR are implemented in the same manner.

From Figs. 6 and 7, we see that CFFR correctly classified the test sample, whereas the global version of CFFR failed to do so. Figs. 8–10 show the experimental results on the ORL, Feret and AR databases, respectively. In these figures, the vertical axis

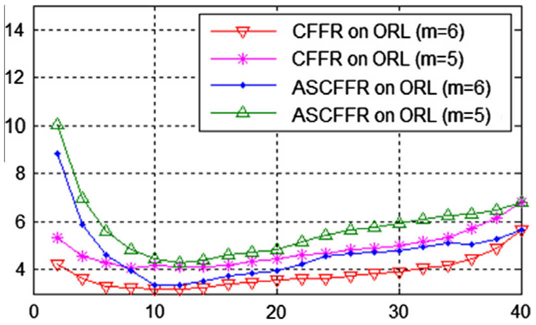

Fig. 8. Means of the rates of classification errors (%) of CFFR and ASCFFR on the ORL face database. When the value of the horizontal axis equals to 40, the corresponding vertical axis indeed shows the means of the rates of classification errors of the global versions of CFFR and ASCFFR. *m* Stands for the number of the training samples per subject.

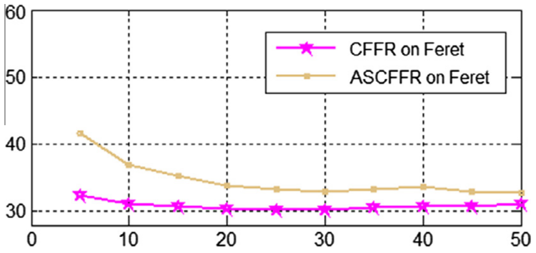

Fig. 9. Means of the rates of classification errors (%) of our method on the Feret database.

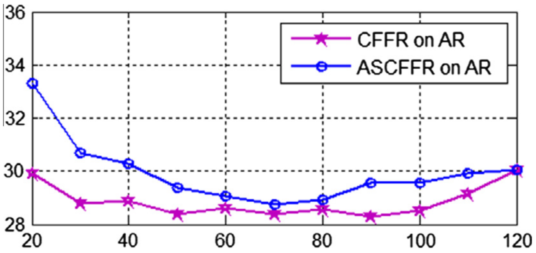

Fig. 10. Means of the rates of classification errors (%) of our method on the AR database. When the value of the horizontal axis equals to 120, the corresponding vertical axis indeed shows the means of the rates of classification errors of the global versions of CFFR and ASCFFR.

shows the mean of the rates of classification errors (%) on all the testing sets and the horizontal axis shows the number of the classes used for classification.

Tables 1–3 show the mean of the rates of classification errors (%) of the nearest neighbor classifier, the minimum and maximum means of the rates of classification errors (%) of CFFR and ASCFFR on the three face databases. For CFFR and ASCFFR on the ORL and AR databases, the tables show only the results of the experiments in which CFFR and ASCFFR take more than fifteen percent of all the classes as the candidate classes, i.e., $s \geqslant 0.15 * L$. These figures and tables clearly show that CFFR (ASCFFR) is able to obtain a much smaller rate of classification errors than the global version of CFFR (ASCFFR). For the ORL and AR databases, we see that our method can also classify more accurately than the nearest neighbor classifier. For the Feret database, the minimum mean of the rate of classification errors obtained using our method is also much lower than that obtained using the nearest neighbor classifier. The figures also show that our method can achieve a very low rate of classification errors when s is set to a proper value.

Tables 4–6 show the classification results of LDA and PCA. As the within-class scatter matrix S_w of LDA was singular, we changed it to $S_w + 0.001I$ where I is the identity. These tables show that CFFR and ASCFFR can classify much more accurately than PCA. For example, while PCA on the Feret database obtained a minimum error rate of 36.02%, CFFR and ASCFFR achieved a minimum error rate of 30.22% and 32.74%, respectively. In most cases, CFFR and ASCFFR on the Feret database obtained a lower error rate than LDA. Moreover, CFFR and ASCFFR on the AR database obtained a much lower error rate than LDA. LDA on the AR database obtained the error rate of 34.21%, whereas CFFR obtained a minimum error rate of 28.29%. The minimum rate of classification errors of CFFR and ASCFFR on the ORL database was also lower than that of LDA.

Table 1
Means of the rates of classification errors of the nearest neighbor classifier, the minimum and maximum means of the rates of classification errors of CFFR and ASCFFR on the ORL database.

Number of training samples per subject	Nearest neighbor classifier (%)	Minimum mean of CFFR (%)	Minimum mean of ASCFFR (%)	Maximum mean of CFFR and ASCFFR (%)	The global versions of CFFR and ASCFFR (%)
6	5.91	3.18	3.37	5.64	5.64
5	7.55	4.07	4.30	6.81	6.81

Table 2
Means of the rates of classification errors of the nearest neighbor classifier, the minimum and maximum means of the rates of classification errors of CFFR and ASCFFR on the Feret database.

Number of training samples per subject	Nearest neighbor classifier (%)	Minimum mean of CFFR (%)	Minimum mean of ASCFFR (%)	Maximum mean of CFFR and ASCFFR (%)	The global versions of CFFR and ASCFFR (%)
4	36.05	30.22	32.74	42.14	42.14

Table 3
Means of the rates of classification errors of the nearest neighbor classifier, the minimum and maximum means of the rates of classification errors of CFFR and ASCFFR on the AR database.

Number of training samples per subject	Nearest neighbor classifier (%)	Minimum mean of CFFR (%)	Minimum mean of ASCFFR (%)	Maximum mean of CFFR and ASCFFR (%)	The global versions of CFFR and ASCFFR (%)
8	41.76	28.29	28.75	33.29	30.05

Table 4
The mean of the classification error rates of PCA and LDA on the AR database. The number in the first row is the number of the transform axes used.

Number of the transform axes used	50 (%)	100 (%)	150 (%)	200 (%)	199 (%)
PCA	45.32	42.69	42.13	41.85	
LDA					34.21

Table 5
The mean of the classification error rates of PCA and LDA on the Feret database. The number in the first row is the number of the transform axes used.

Number of the transform axes used	50 (%)	100 (%)	150 (%)	200 (%)	199 (%)
PCA	38.00	36.38	36.19	36.02	
LDA					36.31

Table 6
The mean of the classification error rates of PCA and LDA on the ORL database. The number in the first row is the number of training samples per subject. The number in second column denotes the number of transform axes used in PCA and LDA, respectively.

Number of training samples per subject		5 (%)	6 (%)
PCA	50	7.18	5.22
	100	7.87	5.96
	150	7.78	6.12
	200	7.55	6.02
LDA	39	4.82	3.71

Table 7
The mean of the classification error rates of the Gabor-filter-based nearest neighbor classifier.

Number of training samples per subject	5 (%)	ORL	Feret	AR
		6 (%)	4 (%)	8 (%)
(Mean of) classification error rate	6.05	4.51	33.60	40.14

We also tested the Gabor-filter-based nearest neighbor classifier. We used nine Gabor transformation results (the Gabor transformation was carried out at three different orientations and scales) of each face image to perform classification. For each of the ORL, Feret and AR databases, we used the same training sets and testing sets presented above. The experimental results are shown in Table 7. We see that CFFR can also obtain a lower error rate than the Gabor-filter-based nearest neighbor classifier.

Our method simply assumes that each subject has an equal number of images. As a result, the algorithm might obtain an unsatisfactory accuracy in the case where different persons provide different numbers of images. However, in order to address the above issue, we can slightly modify our method. In particular, under the condition that different persons provide different numbers of images, we can revise Eq. (8) to $de_r = n_r||y - f_r||^2$, $r \in C$ where n_r denote the number of training samples from the rth class. The above formula allows the residual between the test sample and all the f_r to be evaluated in a fair way by simultaneously exploiting $||y - f_r||^2$and n_r to calculate the residual.

5. Conclusions

The proposed coarse-to-fine face recognition method uses two consecutive stages to classify face images. Its first stage performs a coarse classification, determining a small number of candidate class labels of the test sample. The second stage conducts a fine classification, determining the ultimate class label of the test sample. The proposed method is simple, theoretically reasonable and computationally very efficient. One of the main rationales of this method is that it reasonably makes the classification problem become a simpler one with fewer classes. Moreover, the proposed method can be clearly interpreted. The proposed method obtains a promising accuracy in a large number of face recognition experiments.

Acknowledgements

This article is partly supported by the Key Laboratory of Network Oriented Intelligent Computation, Program for New Century Excellent Talents in University (Nos. NCET-08-0156 and NCET-08-0155), National Nature Science Committee of China under Grant Nos. 61071179, 61203376, 61263032, 61202276 and 61272292, the Fundamental Research Funds for the Central Universities (HIT.NSRIF. 2009130) as well as the ITF fund from the HKSAR Government and the central fund from The Hong Kong Polytechnic University.

References

[1] A. Ahmadi, S. Omatu, T. Fujinaka, T. Kosaka, Improvement of reliability in banknote classification using reject option and local PCA, Information Sciences 168 (2004) 277–293.
[2] A.F. Abate, M. Nappi, D. Riccio, G. Sabatino, 2D and 3D face recognition: a survey, Pattern Recognition Letters 28 (2007) 1885–1906.
[3] A.M. Martinez, A.C. Kak, PCA versus LDA, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (2001) 228–233.
[4] B. Huang, J. Wu, D. Zhang, N. Li, Tongue shape classification by geometric features, Information Sciences 180 (2010) 312–324.
[5] B. Scholkopf, A. Smola, Learning with Kernels, MIT Press, Cambridge, MA, 2002.
[6] B. Scholkopf, A. Smola, K.R. Muller, Kernel principal component analysis, in: Conf. Artificial Neural Networks – ICANN'97, Berlin, 1997, pp. 583–588.
[7] B. Scholkopf, A. Smola, K.R. Muller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Computation 10 (1998) 1299–1319.
[8] C. Feher, Y. Elovici, Robert Moskovitch, Lior Rokach, Alon Schclar, User identity verification via mouse dynamics, Information Sciences 201 (2012) 19–36.
[9] D. Tao, X. Tang, Kernel full-space biased discriminant analysis, in: Conf. ICME, 2004, pp. 1287–1290.
[10] E. Gumus, N. Kilic, A. Sertbas, O.N. Uçan, Eigenfaces and support vector machine approaches for hybrid face recognition, The Online Journal on Electronics and Electrical Engineering 2 (2010) 308–310.
[11] E. Gumus, N. Kilic, A. Sertbas, O.N. Uçan, Evaluation of face recognition techniques using PCA, wavelets and SVM, Expert Systems with Applications 37 (2010) 6404–6408.
[12] G.F. Lu, Y. Wang, Feature extraction using a fast null space based linear discriminant analysis algorithms, Information Sciences 193 (2012) 72–80.
[13] H.J. Li, J.S. Zhang, Z.T. Zhang, Generating cancelable palmprint templates via coupled nonlinear dynamic filters and multiple orientation palmcodes, Information Sciences 180 (2010) 3876–3893.
[14] H. Ryu, J.C. Yoon, S.S. Chun, Sanghoon Sull: coarse-to-Fine classification for image-based face detection, CIVR (2006) 291–299.
[15] H. Tang, H. Maitre, N. Boujemaa, W. Jiang, On the relevance of linear discriminative features, Information Sciences 180 (2010) 3422–3433.
[16] H. Yin, W. Huang, Adaptive nonlinear manifolds and their applications to pattern recognition, Information Sciences 180 (2010) 2649–2662.
[17] J.F. Connolly, E. Granger, R. Sabourin, An adaptive classification system for video-based face recognition, Information Sciences 192 (2012) 50–70.
[18] J. Ruiz-del-Solar, R. Verschae, M. Correa, Recognition of faces in unconstrained environments: a comparative study, EURASIP Journal on Advances in Signal Processing 2009 (2009) 1–20.
[19] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, M. Yi, Robust face recognition via sparse representation, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (2009) 210–227.
[20] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T.S. Huang, S.C. Yan, Sparse representation for computer vision and pattern recognition, Proceedings of the IEEE 98 (2010) 1031–1044.
[21] J. Yang, D. Zhang, A.F. Frangi, J.Y. Yang, Two-dimensional PCA: a new approach to appearance-based face representation and recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (2004) 131–137.
[22] J. Yang, J.Y. Yang, A.F. Frangi, Combined fisherfaces framework, Image Vision Computing 21 (2003) 1037–1044.
[23] K.R. Muller, S. Mika, G. Rätsch, K. Tsuda, B. Schölkopf, An introduction to kernel-based learning algorithms, IEEE Transactions on Neural Network 12 (2001) 181–201.
[24] M.E. Tipping, Sparse kernel principal component analysis, in: T.K. Leen, T.G. Dietterich, V. Tresp (Eds.), NIPS 2000: Neural Information Processing Systems, MIT Press, 2000, pp. 633–639.
[25] M. Kirby, L. Sirovich, Application of the KL phase for the characterization of human faces, IEEE Transactions on Pattern Analysis and Machine Intelligence 12 (1990) 103–108.
[26] M.T. Harandi, M.N. Ahmadabadi, B.N. Araabi, Optimal local basis: a reinforcement learning approach for face recognition, International Journal of Computer Vision 81 (2009) 191–204.

[27] M. Turk, A. Pentland, Eigenfaces for recognition, Journal of Cognitive Neurosicence 3 (1991) 71–86.

[28] M. Sugiyama, Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis, Journal of Machine Learning Research 8 (2007) 1027–1061.

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

[30] P.J. Phillips, The Facial Recognition Technology (FERET) Database. <http://www.itl.nist.gov/iad/humanid/feret/feret_master.html>.

[31] P. Melin, D. Sánchez, O. Castillos, Genetic optimization of modular neural networks with fuzzy response integration for human recognition, Information Sciences 197 (2012) 1–9.

[32] R. Chellappa, C.L. Wilson, S. Sirohey, Human and machine recognition of faces: a survey, Proceedings of the IEEE 83 (1995) 705–740.

[33] S. Gangaputra, D. Geman, A design principle for coarse-to-fine classification, in: Conf. Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and, Pattern Recognition, CVPR'06, 2006, pp. 1877–1884.

[34] The AR Database of Faces. <http://www.cobweb.ecn.purdue.edu/∼aleix/aleix_face_DB.html>.

[35] The ORL Database of Faces. <http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>.

[36] T.Y. Changs, Dynamically generate a long-lived private key based on password keystroke features and neural networks, Information Sciences 211 (2012) 36–47.

[37] V. Vural, G. Fung, B. Krishnapuram, J.G. Dy, B. Rao, Using local dependencies within batches to improve large margin classifiers, Journal of Machine Learning Research 10 (2009) 183–206.

[38] W.K. Yang, C.Y. Sun, L. Zhang, K. Ricanek, Laplacian bidirectional PCA for face recognition, Neurocomputing 74 (2010) 487–493.

[39] W. Zhao, R. Chellappa, A. Rosenfeld, P.J. Phillips, Face recognition: a literature survey, ACM Computing Surveys (2003) 399–458.

[40] Y. Xu, A. Zhong, J. Yang, D. Zhang, LPP solution schemes for use with face recognition, Pattern Recognition 43 (2010) 4165–4176.

[41] Y. Xu, D. Zhang, F.X. Song, J.Y. Yang, Z. Jing, M. Li, A method for speeding up feature extraction based on KPCA, Neurocomputing 70 (2007) 1056–1061.

[42] Y. Xu, D. Zhang, J. Yang, J.Y. Yang, A two-phase test sample sparse representation method for use with face recognition, IEEE Transactions on Circuits and Systems for Video Technology 21 (2011) 1255–1262.

[43] Y. Xu, D. Zhang, J.Y. Yang, A feature extraction method for use with Bimodal biometrics, Pattern Recognition 43 (2010) 1106–1115.

[44] Y. Xu, G. Feng, Y. Zhao, One improvement to two-dimensional locality preserving projection method for use with face recognition, Neurocomputing 73 (2009) 245–249.

[45] Y. Xu, Z. Jin, Down-sampling face images and low-resolution face recognition, in: Conf. The Third International Conference on Innovative Computing, Information and Control, Dalian, China, 2008, pp. 392−395.

[46] Y. Zeng, Y. Yang, L. Zhao, Nonparametric classification based on local mean and class statistics, Expert Systems with Applications 36 (2009) 8443–8448.

[47] Z.Y. Liu, K.C. Chiu, L. Xu, Improved system for object detection and star/galaxy classification via local subspace analysis, Neural Networks 16 (2003) 437–451.